REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* 08-07-2010	**2. REPORT TYPE** Final Report	**3. DATES COVERED** *(From – To)* 13-Apr-09 - 08-Jul-10

4. TITLE AND SUBTITLE Research of multimodular recurrent neural networks for providing adaptive control of complicated dynamical objects	**5a. CONTRACT NUMBER** STCU Registration No: P-357
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Alexander Michaylovich Reznik	**5d. PROJECT NUMBER**
	5d. TASK NUMBER
	5e. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Institute of Mathematical Machines and Systems, Academician Glushkov 42, Kiev 03187, Ukraine	**8. PERFORMING ORGANIZATION REPORT NUMBER** N/A
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) EOARD, Unit 4515 BOX 14, APO AE 09421	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) STCU 08-8008

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This report results from a contract tasking Institute of Mathematical Machines and Systems as follows: Main target of research are dynamic recurrent neural networks and their application for adaptive control of complicated dynamical systems. Recurrent neural networks, unlike classical methods, don't require full a priori information about properties of controlled object. So their use allows to achieve high precision and reliability for control of complicated dynamical objects in conditions of lack of information about internal state of the object.

Distinctive feature of researched in this project recurrent neural networks is the use of dynamical neurons, that are able to ½forget╖ outdated information, and multimodular architecture, that allows to flexibly combine adaptation for changing environment properties and features of associative memory that keeps different controlled object's models. Neural associative memory is able to recognize large-scale changes of environment conditions, that allows to reconfigure network's structure in real time for more efficient control.

Project includes the critical review of publications in the given area, theoretical research of training methods of dynamical recurrent neural networks, methods of reconfiguration of such networks, decomposition of neurocontrolling task, and experimental test of achieved theoretical results, that will be performed on MNN CAD with the use of specially developed models of RNN and controlled objects.

15. SUBJECT TERMS

EOARD, Mathematical And Computer Sciences, Mathematical And Computer Sciences

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18, NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT UNCLAS	**b. ABSTRACT** UNCLAS	**c. THIS PAGE** UNCLAS	UL	41	JAMES LAWTON Ph. D.
					19b. TELEPHONE NUMBER *(Include area code)* +44 (0)1895 616187

Research of recurrent dynamic neural networks for adaptive control of complex dynamic systems

Project manager: Reznik Olexandr Mykhaylovich, Full Doctor of Science (Technical Science)
Tel.: +380 44 5265548, Fax: +380 44 5266457, E-mail: neuro@immsp.kiev.ua
Organizations: Institute of Mathematical Machines and Systems Problems of National Academy of Sciences of Ukraine
Funding party: European Office of Aerospace Research and Development Project
Start date: 01.05.2009
Project duration: 1 year
Reported Period (#3): 01.05.2009– 01.05.2010
Submission date: 01.05.2010

Summary

The report contains final results of the project P-357: "Research of recurrent dynamic neural networks for adaptive control of complicated dynamic systems". Recurrent neural networks have unique dynamic properties for application in neurocontrol systems. But their training has high computational complexity and the training process is non-stable, so these problems prevent wide application of such networks for neurocontrol tasks. We studied ways to solve these problems in the project by applying multimodular architecture and by developing of new methods of training of recurrent neural networks. Multimodular architecture is widely used in powerful recognition systems based on feedforward neural networks. In these systems processing of dynamic data is performed by using additional tapped delay lines in network's input. Increasing the recognition rate can be done by providing specialization of modules by training them on different parts of data. Such fragmentation of data during training process may interrupt consistency of data in a memory of neural network that is not acceptable during the control process of dynamic objects. Therefore, studying of problems of using of multimodular neural networks for adaptive control of complex objects, providing the review of existing decomposition methods of control systems and finding new ways of making considerable decrease of the computational complexity of the training methods of such networks are important research directions.

Research activities related to this project were performed in three directions: 1) theoretical analysis of possibility of speeding up the training process of recurrent neural networks and developing the new model of Dynamic Associative Memory; 2) development of multimodular recurrent neural network structure and methods for its training on problem of adaptive control of complicated objects; 3) development software package for modeling of recurrent neural networks and performing the experimental study of the control of complicated dynamic systems. Detailed materials related to this research are presented in intermediate reports TO01-TO03.

Results of our research related to the first direction are presented in paragraph 2. New concept of Open Recurrent Dynamic Network that allows dramatically decrease computational complexity of the training process by using non-iterative methods is proposed. Theoretical study was made, mechanisms of dynamic attractor's forming and degradation and ability for controlling of their behavior were shown for this network. Software model of neural Dynamic Associative Memory was developed and theoretical estimations of it associative properties were experimentally checked.

Paragraph 3 contains results related to the second direction where possible schemes of decomposition were considered, alternative approaches for multimodular organization were analyzed with using committees of experts and by decomposition of controlled object. Methods of direct and inverse neurocontrol and possibilities of their implementation on the base of recurrent perceptrons and Dynamic Associative Memory were considered. Experimental program related to modular organization of neurocontrol was developed. Library of dynamic processes for experiments was created.

Paragraph 4 contains results of the third direction that includes developing the original experimental tool "Neuroconveyor" for multimodular recurrent networks study. Software tool structure is optimized for problems of control of dynamic processes. Data flow and the order of performed operation are programmed by using XML-based language in the configuration file. During processing of the file, corresponding structure of neural networks

modules is created in the memory, and required operations are performed on it. Modules includes MLP network, recurrent MLP network, Elman’s network, Dynamic Associative Memory and specialized module for control of dynamic objects in real-time that are imported from SIMULINK package.

In Paragraphs 5 – 7 experiments that were done at last project stage are presented. Paragraph 5 contains results of experimental series related to organized as experts committees multimodular neural networks on tasks of modeling stochastic (Wolf Numbers) and algorithmic pseudo-stochastic (Mackey-Glass Process) dynamic sequences. In Paragraph 6 experimental test of “Neuroconveyor” system on 1D and 2D versions of inverted pendulum stabilization tasks are presented. Paragraph 7 contains experimental results related to using multimodular neural networks for hand gestures recognition task.

Introduction

Efficient device and process control is a crucial scientific and engineering problem. Growing requirements for real-time control and complication of controlled objects leads to the quest for novel approaches to this domain, e.g. usage of recurrent neural networks. Such networks can learn to recognize sequences of stimuli and process data streams in real time. Known studies of recurrent neural networks (RNNs) confirm their ability to control dynamic systems and processes. But most of these studies deal wet relatively simple networks, only with few neurons. Currently there is no direct evidence that we can extend these results to more complex neural networks and real-life (controlled) objects. Very high resource consumption and learning volatility of large recurrent NNs are the main issues here. Existing studies of speed-up technique for recurrent NN learning did not give significant results, so application of these network for complex dynamic control is still an open problem.

Project P-357 "Research of recurrent dynamic neural networks for adaptive control of complex dynamic systems" is intended to solve this problem by means of usage of multimodular architectures of recurrent neural networks and enhancement of their learning algorithms.

Multimodular architecture is popular in modern recognition systems based on backpropagation NNs. Efficiency increases there due to neural module specialization: different modules are trained on different subsets of the training sample. Hierarchy in decision making serves the same purpose. On the contrary, in recurrent NNs, that have their own dynamics, training sample segmentation may affect network's memory content and break entire control process. Therefore during the project we had to revise methods of modular organization of neural control and to explore new ways of resource saving for recurrent NN learning. We studied alternative approaches for modular architectures "committees of experts" paradigm and by decomposition of controlled object into autonomous subsystems. Existing learning methods were considered and new paradigm of Open Recurrent Neural Network what can dramatically solve this problem was proposed. We elaborated theory of these networks and proposed non-iterative learning techniques for them.

Besides theoretical studies we performed experimental research of multimodular recurrent networks for dynamic object control. For this purpose the dynamic model library was developed, as well as experimental software package of multimodular recurrent NNs, they were tested in neural control tasks. Some experiments were performed in MATLAB framework using the 4-noded cluster each of whose nodes has two 4-core processors Intel Xeon QuadCore and 16 GB of RAM.

In scope of the project we obtained novel results in multimodular recurrent and their applications to complex dynamic system control; the novel model of dynamic associative memory for neural control was created. We developed the experimental software tool "Neuroconveyor" for real-time dynamic control experiments and proposed some promising directions of studies in this domain.

Methods, Assumptions, and Procedures

1. Goal of the project and main research directions

Goal of the project P-357 is to find methods of using recurrent neural networks for design of efficient control of complicated dynamic objects. Achieving this goal needs for considerable decreasing of resources that are needed for training recurrent neural networks and improving architecture design of recurrent neural networks and neurocontrol systems. In this project, there were studied multimodular neural networks and based on them control systems. Research was performed in the following directions:

- Theoretical analysis of decreasing of computational complexity of the training process of recurrent neural networks and developing the new model of Dynamic Associative Memory;
- Development of multimodular recurrent neural network structure and methods for its training on the problem of adaptive control of complicated systems;
- Development of a software tool for experimental study of multimodular recurrent neural networks on the problem of control of complicated dynamic objects.

Works related to this project were performed during 12 months in 4 stages, by 3 months for each stage. Detailed description of results obtained during first three stages is presented in quarter reports TO0 – TO03. This final report contains brief explanation of results about main directions of research related to the first three stages (Paragraph 2-4) and results of experiments provided on the final 4-th stage (Paragraph 5-7).

2. Theoretical analysis and development of new model of Dynamic Associative Memory

High computational cost of training of recurrent networks is the main problem that prevents their practical application. During two last decades, there were many studies dedicated to this problem research, but there isn't solution that is efficient enough. In the research that was performed according to the project, a new approach is proposed. The study includes IJCNN conference proceedings, Neural Networks journal and Neurocomputing journal publications review on recurrent networks problematic for 1999-2009 years. The result is published in [1]. It was shown that most of researches were dedicated to recurrent multulayer perceptrons (RMLP), that were trained by back propagation through time method (BPTT). Networks with one hidden layer of neurons were studied at most cases. Significantly lower number of publications was dedicated to Elman's network, particularly for its practical application. In theoretical researches the most popular was dynamic version of the Hopfield's network. Significant part of these studies was dedicated to the dynamic network's stability research. Many researchers focused on decreasing of the training time. The best perspectives in this field have hybrid multimodular networks and echo state network (ESN). Hybrid networks are based on the principle of data granulation around the centers of radial basis functions (in RBF networks) or winning neurons (Kohonen's maps). Such granulation changes time-spatial metrics of data that allows decomposition of input processes on relatively independent subprocesses that could be processed with parallel-consequent scheme. Division on subprocesses could be made during system development, or automatically, using a self-training process. Multimodular neural networks, are created during this process, could be treated like committees of experts, could use different neuroparadigms, and different levels of decision making could be used. Hybrid multimodular structure is efficient in many applications, but doesn't solve the problem of high computational cost of recurrent networks training. The most radical solution was proposed by Jagger in 2001. His ESN system contains two modules – recurrent network with fixed connections, and perceptron with weights, that are adjusted by training. Due to fixing recurrent connections, training occurs much faster. But the result isn't very significant, because total network's performance depends on reservoir part that has random behavior and isn't modified during training.

As the result of the performed analysis, it was concluded that for adaptive control of complicated objects multimodular neural networks that acts like experts committee that could have hierarchical structure. But existing methods doesn't solve main problem of recurrent networks – the high computational complexity of their training, so future study was focused on finding for new ways of approaching this problem. At the beginning, the hypothesis of a special role of internal recurrent connections of neurons was studied, that is based on analogy with previously discovered effect of desaturation of associative memory in Hopfield's network. Performed experiments didn't prove such analogy, so the direction of the research was changed, and open recurrent networks, that is similar to the Hopfield's network, was proposed, that allowed apply non-iterative training methods, that radically decreased computational complexity. Unlike the Hopfield's network, Open Recurrent Neural Network (Fig.1) has an additional set of delayed connections. Its state is open for external observation, that allows mathematical solution of its stable state, and so dynamic attractors could be defined, and corresponding weights could be determined. The theory of such network was introduced, where its dynamical attractors are represented as sequences of static attractors, and non-iterative method for its training is proposed. Estimations of memory size and attraction radius were obtained, and possibility of control of network's state was proven. Idea of this control is illustrated on Fig. 2, where an external input controls transition between static frames of the network's dynamic attractor.

For experimental verification of obtained results a software model of the dynamic associative memory was developed. It was implemented as specialized module for multimodular recurrent networks system, developed for this project. Fig. 3 illustrates an example of dynamic associative memory application for recognition of dynamic object, represented as moving circle. Intersection of trajectories of two dynamic attractors is shown. Fig. 4 contains theoretical and experimental dependencies of attraction radius on memory fill (amount of stored dynamic attractors).

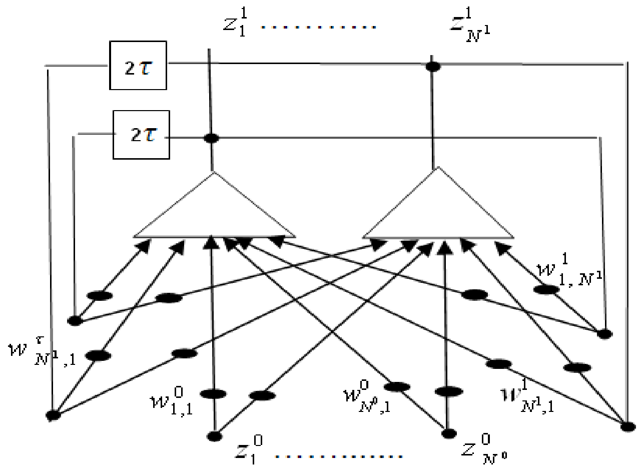


Fig.1. Dynamic associative memory

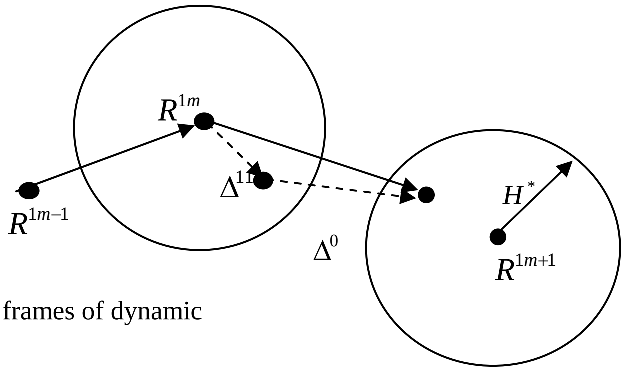
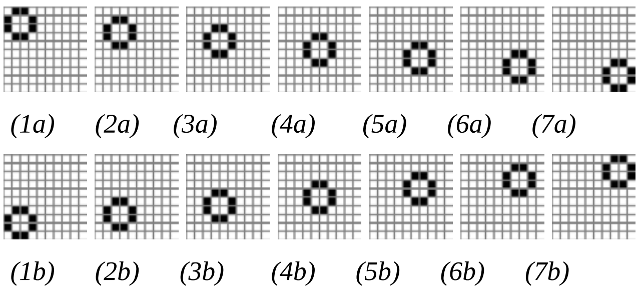
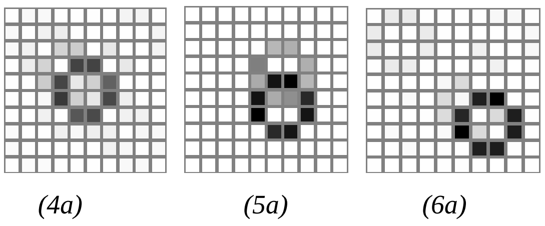


Fig.2. Transition between frames of dynamic attractor.

Results of this study are described in detail in reports TO01, TO03, and in [2, 3]. Experimental verification of theoretical conclusions allows to consider that dynamic associative memory has perspectives in application for control of complicated dynamic objects.





Postsynaptic potentials when reproducing attractor (a)

Fig.3. Dynamic attractors with one intersecting frame (4a=4b).

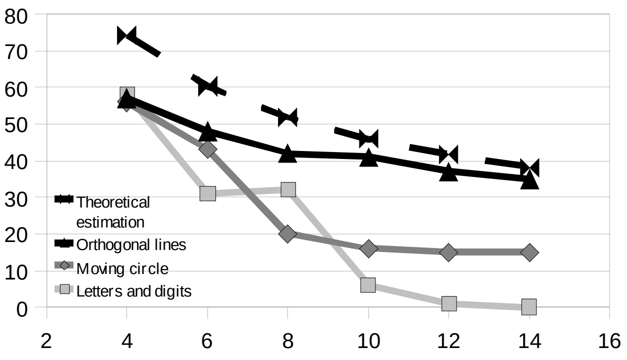


Fig.4. Dependency of attraction radius on number and type of memorized vectors.

3. Development of multimodular recurrent neural network structure and methods of its training on problem of adaptive control of complicated objects

Using multimodular recurrent neural networks for dynamic processes control task needs appropriate organizing of neurocontrol process. We have analyzed current state of research in this area, considered possible schemes of neurocontrol system decomposition and modular structures, developed experimental program, prepared base for providing these experiments. Most difficult problems neurocontrol area are related to development of inverse model that must to take into account non-linear nature of object together with inversion of casual effect relations that defines object’s behavior. To solve these problems, neurocontrol problem can be separated to two sub-problems: preformed by neuroemulator object modeling and performed by neurocontroller adaptive control. Such separation allows present inverse dynamics of object more precisely and make less sensitive to noise in time of adaptive control of the object. Neuroemulator depending on training process organization can realize direct or inverse model of the object. Neurocontrol called “direct” or “inverse” accordingly.

Alternative approaches to multimodular organization of neurocontrol were considered: on the base of hierarchical committee of experts or on the base of decomposition of controlled object on subsystems with autonomous control systems. Using direct and inverse neurocontrol studied. Recurrent perceptron was shown to be the most appropriate neural network for both models. Scheme of direct neurocontrol with using neuroemulator and neurocontroller on the base of recurrent perceptron is presented on Fig. 5. Unlike of recurrent perceptron, Dynamic Associative Memory may be used as neuroemulator in inverse scheme only. Adaptive neurocontrol with inverse neuroemulator on the basis of Dynamic Associative Memory is shown on Fig. 6.

Recurrent neural networks can to be multimodular. Multimodular architecture of feedforward recurrent neural neural network (recurrent perceptron) was developed and modules structure was defined. It is presented on Fig. 7.

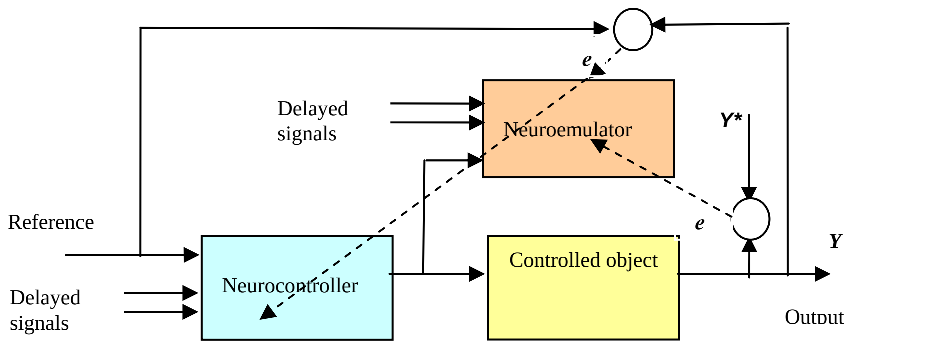


Fig. 5. Adaptive neurocontrol scheme. Y –controlled object output; Y*- estimated object output; e- error of object output

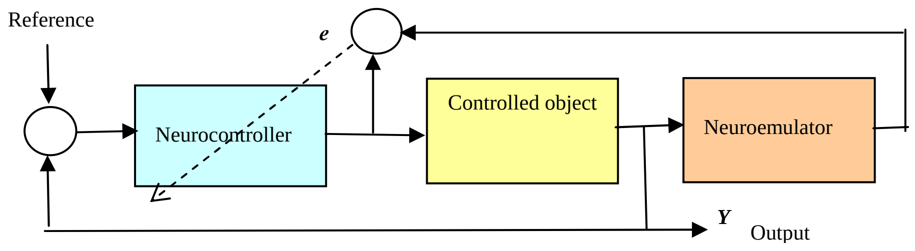


Fig. 6. Inverse neurocontrol, neuroemulator training regime

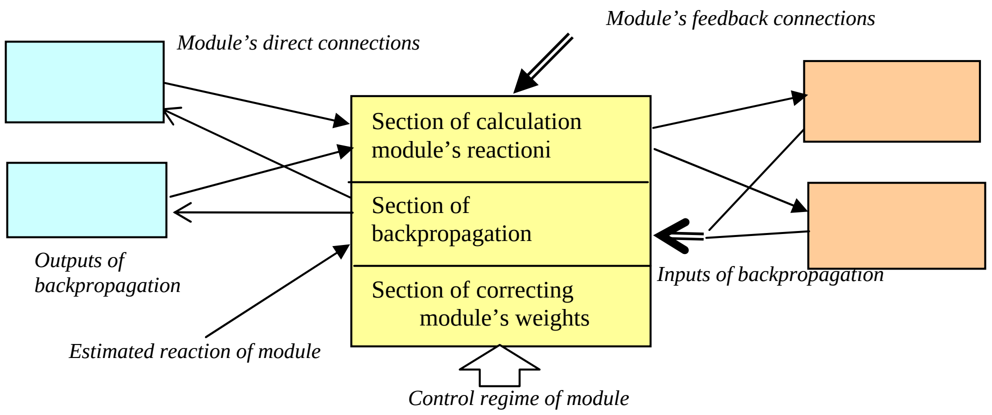

Module's feedback connections
Module's direct connections
Section of calculation module's reactioni
Section of backpropagation
Section of correcting module's weights
Outputs of backpropagation
Inputs of backpropagation
Estimated reaction of module
Control regime of module

Fig. 7. Structure of module of feedforward recurrent neural network

4. Development of the software system for multimodular recurrent neural networks modeling

The experimental software system for multimodular recurrent neural networks modeling was developed during first three stages of the project. The software system has unique structure that allows the control of dynamic processes, with XML-based language for system structure programming. Structure of the neural networks modules and their working modes are defined into the configuration file. Dynamically linked libraries are loaded during configuration file processing, and multimodular system, illustrated on Fig. 8. is created in memory. There were developed modules for data processing, neural networks modules, and API module for dynamic objects control that allows importing of models from SIMULINK package. There are implemented several recurrent networks, such as recurrent MLP, Elman's network, dynamic associative memory. The recurrent MLP module contains the new improved algorithm for iterative learning. Detailed description of this algorithm is presented in the TO02 report. Fig. 9 illustrates the results of comparison of the algorithm on the Wolf numbers forecasting problem.

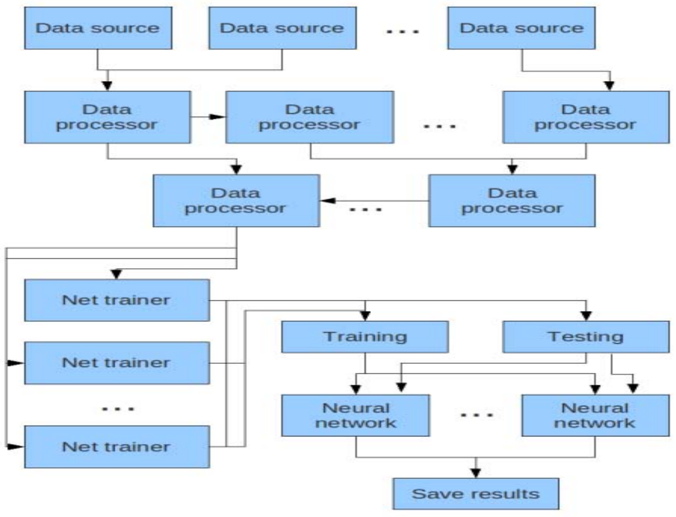


Fig. 8. “Neuroconveyor” system structure.

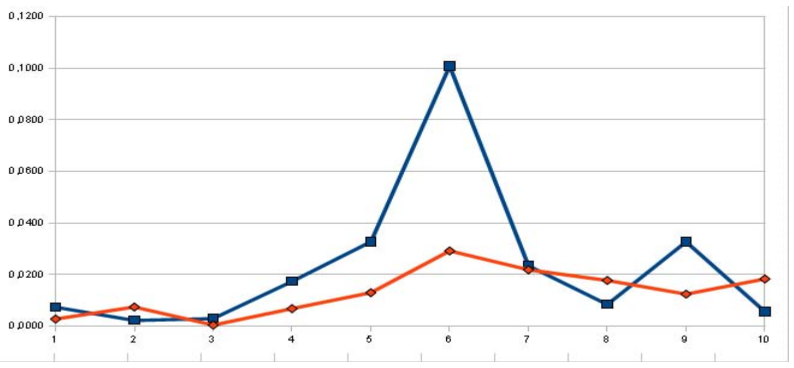


Fig. 9. Wolf Numbers forecasting errors for the recurrent perceptron, trained by original RTRL method (blue color) and by modified RTRL method (orange color) for different forecasting periods.

For experimental study of multimodular recurrent neural networks and control system, are based on them, it was developed specialized system that contains:

- “Neuroconveyor” kernel;
- 5 modules for data processing (in the form of DLL libraries);
- 4 neural network's modules (in the form of DLL libraries);
- 3 modules of dynamic objects, that are imported from SIMULINK package (in the form of DLL libraries);
- Subsystem for control of dynamic objects and real-time neural networks training.

Fig. 10 illustrates the structure of this system, and Fig. 11 contains screenshot of graphical user interface.

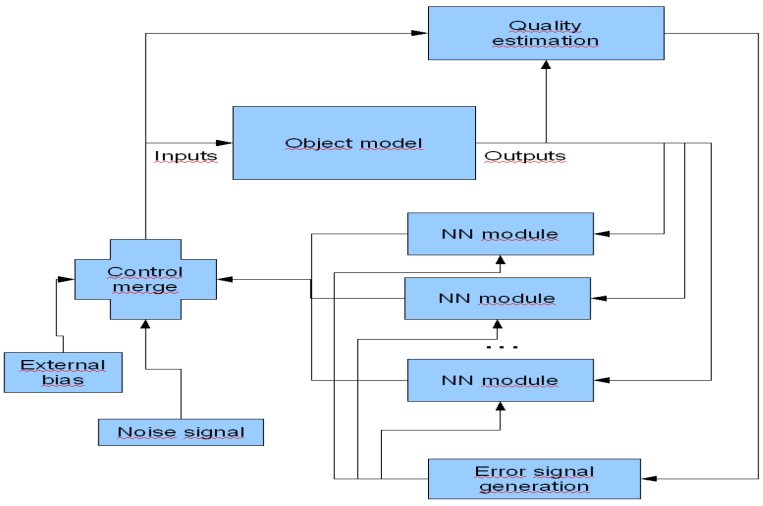


Fig. 10. Multimodular neurocontrol structure.

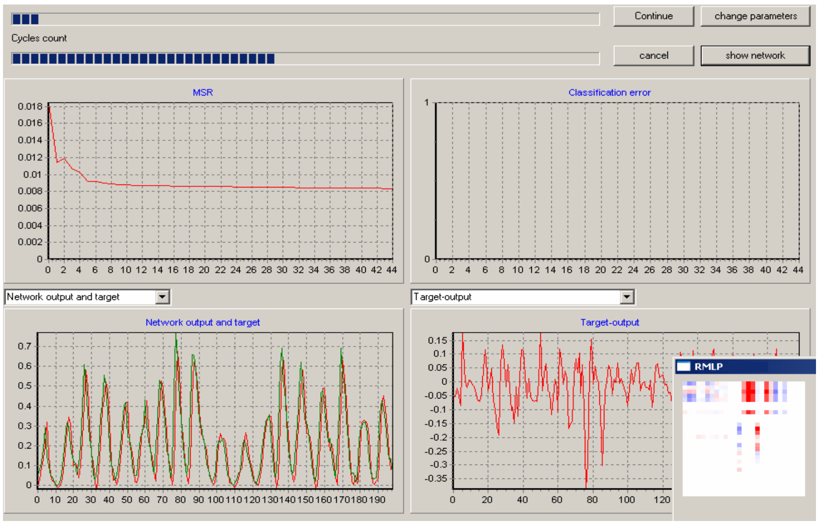


Fig. 11. Graphical user interface of “Neuroconveyor” system (Wolf numbers forecasting problem).

Software system was developed for OS Windows, but system-dependent operations are used only in I/O modules. In general, system has minimal dependence on operational system and could be easily ported on OS UNIX family, or more specific platforms, such as GPU computations environment (for example, CUDA). All modules are implemented using pure C++ without using any specific libraries. The kernel of the “Neuroconveyor” system contains XML-based language parser that is used for system operations programming. Detailed description of structure and implementation of “Neuroconveyor” system could be found in the TO02 report.

5. Experimental research of "comittiees of experts"-like multimodular neural systems

5.1 Description of experiments

To compare different variants of neurocontrol, we reviewed famous tests and testing methods of recurrent neural networks and developed program of experiments. We created Library of dynamic processes that includes the following 7 models:

- Wolf Numbers;
- Mackey-Glass Process;
- Magnet Levitation System;
- Inverted Pendulum;
- Inverted Pendulum 2D;
- Two-unit Vertical Bob;
- Gestures Recognition task.

First four models in the list are 1D dynamic systems that were collected to investigate multimodular systems like mixtures of experts. Residuary models are complex dynamic systems oriented to investigate methods of neurocontrol decomposition. Last two models are original, developed specially for this project.

Experiments on estimation of general possibility of recurrent neural networks to model complex dynamic processes were provided with two famous sequences of data: Wolf Numbers and Mackey-Glass Process. First sequence is records of sunspots by years made by astronomers that were obtained starting from 1700 year. It includes 307 values. It was divided on three parts: 45% (140 values) – for training (TRAIN Data), 20% (60 values) for testing (TEST Data) and 35% (107 values) for testing generalizations properties of predictors (VALIDATION Data).

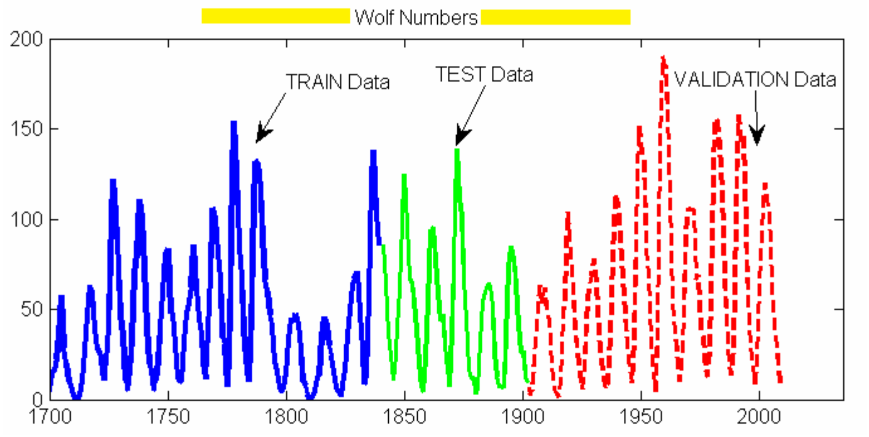


Fig. 12. "Wolf Numbers" sequence for training and testing quality of single and multimodular neural networks-predictors.

"Mackey-Glass Process" sequence is solution of ordinary nonlinear equation with time delay, defined as:

$$\frac{dy(t)}{dt} = \frac{ay(t-\tau)}{1+y(t-\tau)^{10}} - by(t)$$

We used the following parameter values in this equation: $a = 0.2, b = 0.1, \tau = 17$.

For providing honest comparison of forecasting results between these two sequences, Mackey-Glass sequence that was taken has also 307 values, that was divided to train, test and validation sequences with the same number of values: 140, 60 and 107. Values of both sequences were normalized to interval (-1;1).

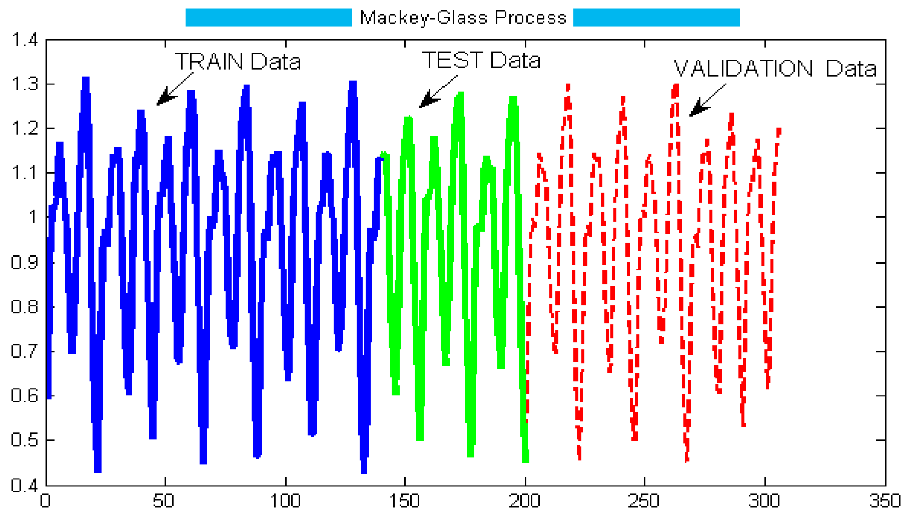


Fig. 13. "Mackey-Glass Process" sequence for training and testing quality of single and multimodular neural networks-predictors.

Forecasting task definition. Forecasting task can be formulated as follows: by known current value of sequence $y(k)$ and few last historical values of this process $y(k-1), y(k-2),..., y(k-m)$ make estimation of the most probable next value $y^*(k+1)$. Estimation procedure repeats for all sequence points that have next values. Number of history values that are taken into account is defined by concrete predictor.

Neural network predictors should be trained on TRAIN part of sequence. Then, the best predictor should be selected by testing it's ability to forecast next values on TEST part of sequence. Criteria of quality was Mean Squared Error, defined as:

$$Error = \frac{1}{N}\sum_{n=1}^{N}(y(n) - y^*(n))$$

where: N – number of process values, $y(n)$ – real process values, $y^*(n)$ – forecasted values. Single or multimodular predictor that has the least error on TEST sequence, takes part in additional test on VALIDATION sequence. Validation result is the final result that characterizes the quality of obtained predictor.

5.2 Neural network types

Linear Neural Network, LNN, or Adaline [4] is static feedforward neural network. Network's output is linear transform of input signal, that is defined by weights vector $\vec{w}$. Weights vector size is proportional to size of input vector.

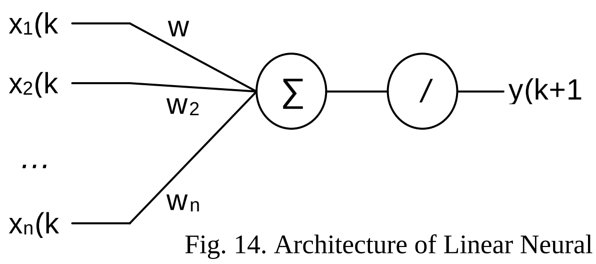


Fig. 14. Architecture of Linear Neural Network, LNN

Multilayered Perceptron Neural Network, MLPNN [5] is non-linear static feedforward neural network. We used 3-layer neural network architecture with one hidden layer and "hyperbolic tahn" activation functions.

Number of neurons in input and output layers are defined by size of input and output vectors. Number of neurons in hidden layer is variable value, it's optimal value depends on input data and can't be defined a-priori. In our report this parameter is marked as *HiddenN*.

Dynamic Linear Neural Network with tapped delay line, DLNN [6] is dynamized version of Linear Neural Network. To provide dynamic properties in this static neural network, Tapped Delay Line was added network's input. It allows to put into network's input not only current value of the process, but also few latest historical values. Number of historic values that should be taken when network works is marked as *InputDelayN*. Architecture of DLNN network is shown on Fig. 15, left.

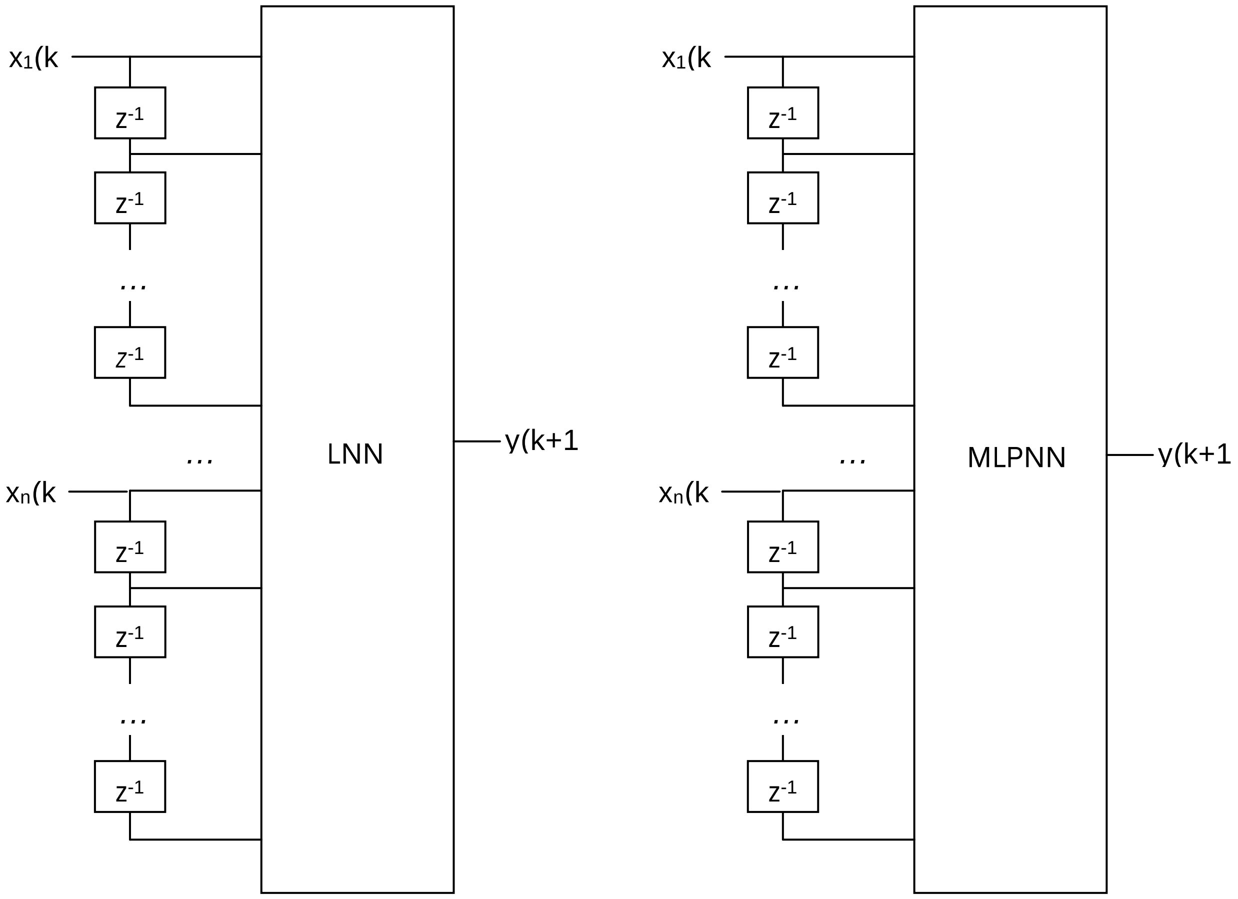


Fig. 15. Architecture of Linear Neural Network with Tapped Delay Line, DLNN (left) and Multilayered Perceptron with Tapped Delay Line, FTDNN (right).

Multilayered Perceptron with Tapped Delay Linem FTDNN (it is also known as Focused Time-Delay Neural Network) is dynamized version of Multilayered Perceptron . Architecture of FTDNN is shown on Fig. 15, right. Number of latest historical values that are used in network input is marked as *InputDelayN*, number of neuron in hidden layer – *HiddenN*.

Recurrent Nonlinear Autoregressive Neural Network with eXogenous inputs, NARXNN [7] is shown on Fig. 16. Structure of this neural network is similar to Multilayered Perceptron with Tapped Delay Line, but it also has additional recurrent connections from network's output to it's input through another Tapped Delay Line. So, this network received not only delayed input values, but delayed own previous predictions as well. Number of delays in this new additional Tapped Delay Line is marked as *OutputDelayN*. Parameters of network *InputDelayN* and *HiddenN* has the same sense as in mentioned Multilayered Perceptron with Tapped Delay Line.

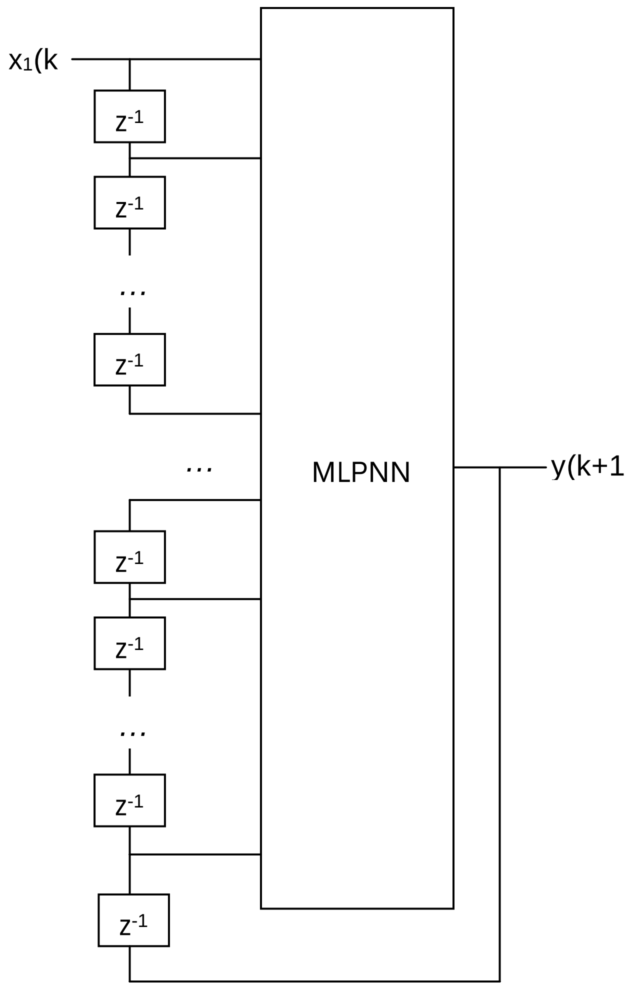


Fig. 16. Architecture of Nonlinear Autoregression Neural Network with eXogenous inputs, NARXNN

5.3 Training Single Neural Networks

For solving the forecasting task that was defined in section 4.1.2, we used both static and dynamic neural networks. All neural networks were trained on the same data sequences, Wolf Numbers and Mackey-Glass Process. First, training all neural networks architectures has been provided on section “TRAIN”. Few iterations of training with different random weight coefficients on the start on the training process were made for all architectures of neural networks, except of LNN and DLNN that were trained by using non-iterative training algorithm. Neural networks that provided the best performance on TEST sub-sequences were also tested on VALIDATION part of sequences. This value is the final neural network’s performance value.

Single Static Linear Neural Network. Least Mean Squares (LMS) algorithm of weights correction was used for training process. Number of training attempts: 1. Total networks trained for every dynamic process: 1.

Table 1. Results of linear static predictions on both sequences

Process.	TRAIN	TEST	TRAIN + TEST	VALIDATION
Wolf Numbers	0.01134	0.01042	0.01128	0.02276
Mackey-Glass Process	0.018738	0.019297	0.018906	0.018598

Single Static Multilayered Perceptron. Networks were trained with using Levenberg-Marquardt training algorithm. Maximum number of epochs: 600. Number of neurons in hidden layer, HiddenN: from 5 to 15. Number of training attempts: 10. Total networks were trained for every dynamic process: 110.

Table 2. Results on sequence “Wolf Numbers”

MLP No.	HiddenN	TRAIN	TEST	TRAIN + TEST	VALIDATIO N
36	8	0.01063	0.01032	0.01075	0.02468
17	6	0.01061	0.01062	0.01081	0.02480
56	6	0.01095	0.01081	0.01112	0.02401
7	8	0.01092	0.01086	0.01110	0.02509
38	10	0.01094	0.01086	0.01111	0.02512

Table 3. Results on sequence “Mackey-Glass Process”

MLP No.	HiddenN	TRAIN	TEST	TRAIN + TEST	VALIDATIO N
62	11	0.017549	0.01792	0.01766	0.018866
89	13	0.017548	0.017924	0.017661	0.018854
106	15	0.017548	0.017925	0.017661	0.018854
109	15	0.017544	0.017958	0.017668	0.018865
85	13	0.017527	0.017964	0.017658	0.018854

Single Dynamic Linear Neural Network. Least Mean Squares (LMS) algorithm of weights correction was used for training process. Tapped delay line length, InputDelayN: from 1 to 10. Number of training attempts: 1. Total networks were trained for every dynamic process: 10.

Table 4. Results on sequence “Wolf Numbers”

DLNN No.	InputDelay N	TRAIN	TEST	TRAIN + TEST	VALIDATIO N
1	1	0.00598	0.00592	0.00597	0.01166
2	1	0.00598	0.00592	0.00597	0.01166
5	2	0.00594	0.00614	0.00600	0.01193
6	2	0.00594	0.00614	0.00600	0.01193
19	5	0.00572	0.00617	0.00587	0.01120

Table 5. Results on sequence “Mackey-Glass Process”

DLNN No.	InputDelay N	TRAIN	TEST	TRAIN+TES T	VALIDATIO N
9	9	0.00369	0.003339	0.003585	0.0035374
10	10	0.003668	0.003347	0.003572	0.0035561
8	8	0.003743	0.003367	0.00363	0.0035391
7	7	0.00372	0.003396	0.003623	0.003526
6	6	0.003754	0.003432	0.003657	0.0035685

Single Dynamic Multilayered Perceptron with Tapped Delay Line. Networks were trained with using Levenberg-Marquardt training algorithm. Maximum number of epochs: 600. Number of neurons in hidden layer, HiddenN: from 5 to 15. Tapped delay line length, InputDelayN: from 1 to 10. Number of training attempts: 10. Total networks were trained for every dynamic process: 1100.

Table 6. Results on sequence “Wolf Numbers”

FTDNN No.	InputDelayN	HiddenN	TRAIN	TEST	TRAIN + TEST	VALIDATION
33	4	5	0.00204	0.00400	0.00269	0.01181
38	1	7	0.00229	0.00443	0.00313	0.01360
208	3	5	0.00204	0.00456	0.00300	0.01091
23	3	5	0.00241	0.00460	0.00314	0.01227
236	3	5	0.00227	0.00488	0.00306	0.01041

Table 7. Results on sequence “Mackey-Glass Process”

FTDNN No.	InputDelayN	HiddenN	TRAIN	TEST	TRAIN+TEST	VALIDATION
401	5	12	4.13E-06	6.81E-06	4.94E-06	1.27E-05
329	4	12	3.96E-06	7.06E-06	4.89E-06	9.22E-06
409	4	14	2.42E-06	7.86E-06	4.05E-06	6.86E-06
290	4	11	6.47E-06	8.56E-06	7.10E-06	9.52E-06
457	5	13	3.29E-06	8.73E-06	4.92E-06	8.56E-06

Single Recurrent Nonlinear AutoRegression Neural Network with eXogenous inputs. Networks were trained with using Levenberg-Marquardt training algorithm and backpropagation through time (BPTT) teqnique. Maximum number of epochs: 30. Number of neurons in hidden layer, HiddenN: from 5 to 15. Tapped delay line length, InputDelayN: from 1 to 5. Number of training attempts: 5. Total networks were trained for every dynamic process: 1100.

Table 8. Results on sequence “Wolf Numbers”

NARX NN No.	InputDelayN	HiddenN	OutputDelayN	TRAIN	TEST	TRAIN + TEST	VALIDATION
2528	2	15	1	0.00214	0.00370	0.00297	0.01300
78	4	5	1	0.00248	0.00373	0.00282	0.01241
1556	3	11	2	0.00231	0.00458	0.00316	0.03803
1329	4	10	1	0.00232	0.00462	0.00294	0.01487
87	4	5	3	0.00202	0.00464	0.00299	0.01529

Table 9. Results on sequence “Mackey-Glass Process”

NARXNN No.	InputDelayN	HiddenN	OutputDelayN	TRAIN	TEST	TRAIN+TEST	VALIDATION
703	7	14	1	1.04E-05	7.69E-06	9.62E-06	1.04E-05
804	8	14	1	1.15E-05	1.08E-05	1.13E-05	1.46E-05
669	6	15	1	9.16E-06	1.19E-05	9.98E-06	9.41E-06
1321	6	15	2	8.54E-06	1.35E-05	1.00E-05	8.95E-06
908	9	14	1	9.38E-06	1.41E-05	1.08E-05	1.56E-05
907	5	13	2	1.56E-05	1.43E-05	1.52E-05	1.83E-05

5.4 Training Multimodular Neural Networks

5.4.1 Ensemble Neural Networks

We trained over 4000 single neural networks, they have variable parameters and different architecture paradigms. Generally, they successfully solved sequences forecasting task and proved their ability to model dynamics of complex process. During making the forecast, every neural network used own internal model of dynamic process that appeared during training this neural network. Situations where two or more single neural networks makes errors at different places of the sequence, but they both have good integral quality are not uncommon.

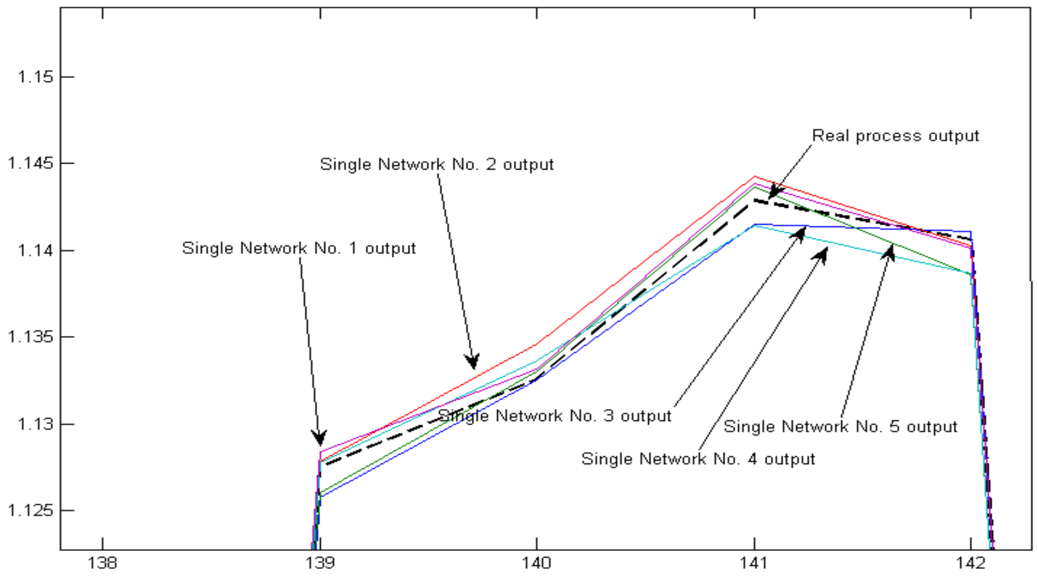


Fig. 17. Forecasts of different single recurrent neural networks and real values of process to forecast

So, we can ask the following question: is it possible to make better the performance of neural prediction system by combining forecasts of trained single neural networks-predictors that provides sufficiently independent results?
To answer on this question, experiments with using multimodular neural network that can be addressed to Ensemble Neural Networks [8] were provided. In these experiments, the real-time forecasts of specially selected single neural networks-predictors were generalized by additional combining modules, g*ating modules* or *gating networks*. We tested different modules as gating modules: simple arithmetic averaging and taking the median and neural networks of different paradigms, that were trained, tested and validated on the same sequences as single networks were. Dividing the sequence into non-intersected parts like it is done Boosting methods for regression [9],[10] is impossible because of short length of used sequences and recurrent nature of single predictors. We used only NARXNN recurrent networks as single neural networks-predictors.

Selection process of single networks into committee has two steps: selection of 20 best networks from 2750 trained recurrent neural networks for every sequence, and selection 5 neural networks form these 20 that have the least value of linear dependency between their outputs. To estimate linear dependence between outputs of single neural networks - candidates to committee, we calculated condition values of network outputs on TRAIN for every set of 5 possible networks from 20 best networks, $C_{20}^{5} = 15504$ sets at all. Condition values were calculated by formulae:

$$cond = \frac{\alpha_{\max}}{\alpha_{\min}}$$

where α – singular values that are square roots of corresponding eigenvalues of correlation matrix between networks outputs. The less condition value is, the less value of linear dependence between networks outputs actually is. For linearly independent vectors condition value is infinity, for linearly dependent outputs it is 0.

Table 10. Committee candidates composed from 5 single recurrent neural networks that have less joint condition value. Number of neural network is network's rating No. in Top-20 best single recurrent neural networks list for sequence "Wolf Numbers"

Variant No.	Condition value	NN 1 rating No.	NN 2 rating No.	NN 3 rating No.	NN 4 rating No.	NN 5 rating No.
1	16.7664002	2	4	13	15	16
2	16.8273275	2	7	12	13	15
3	16.91330968	2	12	13	15	16
4	16.94869564	2	5	12	13	15

Table 11. Committee candidates composed from 5 single recurrent neural networks that have less joint condition value. Number of neural network is network's rating No. in Top-20 best single recurrent neural networks list for sequence "Mackey-Glass Process"

Variant No.	Condition value	NN 1 rating No.	NN 2 rating No.	NN 3 rating No.	NN 4 rating No.	NN 5 rating No.
1	960.4564657	5	10	11	17	20
2	965.3795153	7	15	17	19	20
3	965.7008766	10	11	17	19	20
4	966.7428711	5	10	13	18	20

From tables 10 and 11 it could be seen that single networks that have the best performance, i.e. NNs with rating No. = 1, were not included to the list of best committees. Moreover, every committee has networks with worst rating in Top-20 list.

5.4.2 Multimodular Neural Networks MM

In our experiments, multimodular neural network without using original data in combining module consists of 5 single neural networks-predictors and one combining module. We used committees from 1-st raw at tables 10 and 11 for Wolf Numbers and Mackey-Glass Process accordingly. These committees have the less joint linear dependence by outputs with totally good performance.

Neural networks-predictors in committee, "Predictor NN" receives original data and produces own predictions independently one from another. Combining network, "Gating NN" provides arithmetic averaging, taking the median value or also is neural network. "Gating NN" has to forecast next values of sequence on the basis of forecasts of 5 single neural networks-predictors that are currently in committee.

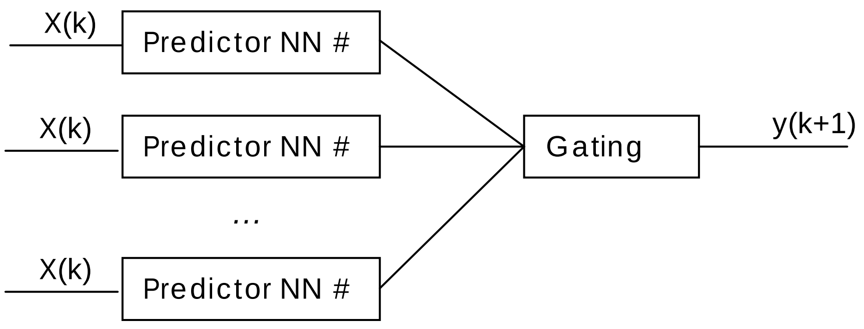


Fig. 18. Architecture multimodular neural network “MM” without using original data in gating module

We used different types of neural networks and simple averaging operations as gating module. It was trained on the same data as single networks were. During this training, weights of networks “Predictor NN” were not changed. In our protocols, this combining model is marked as “MM” (Multi Modular).

Simple variants of gating module of MM multimodular neural network

Table 13. Multimodular neural network with arithmetic averaging as gating module

Sequence name	TRAIN	TEST	TRAIN+TEST	VALIDATION
“Wolf Numbers”	N/A	N/A	0.00262	0.01653
“Mackey-Glass Process”	N/A	N/A	5.63964E-06	9.12857E-06

Table 14. Multimodular neural network with median as gating module

Sequence name	TRAIN	TEST	TRAIN+TEST	VALIDATION
“Wolf Numbers”	N/A	N/A	0.00231	0.01432
“Mackey-Glass Process”	N/A	N/A	5.92201E-06	5.81958E-06

Static Linear Neural Network as gating module of MM multimodular neural network. Least Mean Squares (LMS) algorithm of weights correction was used for training process. Number of training attempts: 1. Total networks trained for every dynamic process: 1.

Table 15. Results with using Linear Neural Network on both sequences

Gating LNN No.	TRAIN	TEST	TRAIN+TEST	VALIDATION
“Wolf Numbers”	0.001283	0.008261	0.003376472	0.019305416
“Mackey-Glass Process”	4.45E-06	1.63E-05	8.0028E-06	4.96907E-05

Static Multilayered Perceptron as gating module of MM multimodular neural network. Networks were trained with using Levenberg-Marquardt training algorithm. Maximum number of epochs: 600. Number of neurons in hidden layer, HiddenN: from 1 to 5. Number of training attempts: 2. Total networks were trained for every dynamic process: 10.

Table 16. Best results on sequence “Wolf Numbers”

Gating MLPNN No.	HiddenN	TRAIN	TEST	TRAIN+TEST	VALIDATIO N
1	1	0.001279	0.008341	0.003397588	0.019326807
2	1	0.001279	0.008341	0.003397565	0.019326662
3	2	0.001214	0.009227	0.003617984	0.022002073
4	2	0.001214	0.009225	0.003617426	0.022007911
5	3	0.001075	0.025955	0.008538917	0.062064543

Table 17. Best results on sequence “Mackey-Glass Process”

Gating MLPNN No.	HiddenN	TRAIN	TEST	TRAIN+TEST	VALIDATION
5	3	9.63401E-07	4.99903E-06	2.17409E-06	0.127941517
6	3	1.41439E-06	9.78928E-06	3.92686E-06	1.21976E-05
8	4	1.70067E-06	1.18485E-05	4.74502E-06	1.52095E-05
7	4	2.18145E-06	1.37543E-05	5.65329E-06	2.40493E-05
4	2	3.3005E-06	1.49845E-05	6.80571E-06	2.48061E-05

Dynamic Linear Neural Network as gating module of MM multimodular neural network. Least Mean Squares (LMS) algorithm of weights correction was used for training process. Tapped delay line length, InputDelayN: from 1 to 10. Number of training attempts: 1. Total networks were trained for every dynamic process: 10.

Table 18. Best results on sequence “Wolf Numbers”

Gating DLNN No.	InputDelayN	TRAIN	TEST	TRAIN+TEST	VALIDATION
1	1	0.001231597	0.008546174	0.00342597	0.022090986
2	2	0.001207638	0.009936213	0.003826211	0.023693571
3	3	0.001165003	0.009996262	0.003814381	0.026555329
4	4	0.001152954	0.010300141	0.00389711	0.028907853
6	6	0.001112897	0.013644866	0.004872487	0.03061912

Table 19. Best results on sequence “Mackey-Glass Process”

Gating DLNN No.	InputDelayN	TRAIN	TEST	TRAIN+TEST	VALIDATION
3	3	3.85456E-06	1.40081E-05	6.90061E-06	4.25028E-05
1	1	4.14892E-06	1.47122E-05	7.31792E-06	5.2564E-05
2	2	4.04556E-06	1.47756E-05	7.26458E-06	4.98061E-05
4	4	3.59324E-06	1.59385E-05	7.29681E-06	6.98237E-05
5	5	3.23055E-06	2.01369E-05	8.30247E-06	9.85357E-05

Multilayered Perceptron with Tapped Delay Line as gating module of MM multimodular neural network. Networks were trained with using Levenberg-Marquardt training algorithm. Maximum number of epochs: 600. Number of neurons in hidden layer, HiddenN: from 1 to 5. Tapped delay line length, InputDelayN: from 1 to 10. Number of training attempts: 2. Total networks were trained for every dynamic process: 100.

Table 20. Best results on sequence “Wolf Numbers”

Gating FTDNN No.	InputDelay	HiddenN	TRAIN	TEST	TRAIN+TEST	VALIDATION

	N					
1	1	1	0.001580308	0.008151096	0.003551544	0.031341897
2	1	1	0.001580301	0.008151106	0.003551542	0.031341685
3	2	1	0.001524004	0.007949587	0.003451679	0.028255113
4	2	1	0.001524003	0.007949591	0.003451679	0.028255083
5	3	1	0.001354562	0.008701716	0.003558708	0.032996802

Table 21. Best results on sequence “Mackey-Glass Process”

Gating FTDNN No.	InputDela y N	Hidden N	TRAIN	TEST	TRAIN+TEST	VALIDATIO N
84	2	5	6.35888E-07	1.05279E-05	3.60348E-06	1.63934E-05
86	3	5	3.81211E-07	1.10229E-05	3.5737E-06	1.83672E-05
82	1	5	5.94683E-07	1.20896E-05	4.04315E-06	1.85975E-05
81	1	5	5.96008E-07	1.21234E-05	4.05422E-06	1.8588E-05
66	3	4	5.14632E-07	2.16459E-05	6.854E-06	1.3824E-05

Recurrent Nonlinear AutoRegression Neural Network with eXogenous inputs as gating module of MM multimodular neural network. Networks were trained with using Levenberg-Marquardt training algorithm and backpropagation through time (BPTT) teqnique. Maximum number of epochs: 30. Number of neurons in hidden layer, HiddenN: from 1 to 5. Tapped delay line length, InputDelayN: from 1 to 5. Number of training attempts: 2. Total networks were trained for every dynamic process: 500.

Table 22. Best results on sequence “Wolf Numbers”

Gating NARXNN No.	InputDelay N	Hidden N	Output DelayN	TRAIN	TEST	TRAIN+TEST	VALIDATIO N
216	2	3	3	0.001179236	0.00672153	0.002841924	0.024983305
224	3	3	2	0.001187459	0.006832144	0.002880865	0.031181848
13	2	1	2	0.001465905	0.00712318	0.003163087	0.033205566
14	2	1	2	0.001464495	0.007125119	0.003162682	0.033165185
201	1	3	1	0.001376383	0.008023668	0.003370569	0.022353417

Table 23. Best results on sequence “Mackey-Glass Process”

Gating NARXNN No.	InputDela y N	Hidden N	Output DelayN	TRAIN	TEST	TRAIN+TEST	VALIDATION
410	1	5	5	1.13236E-05	9.84379E-06	1.08797E-05	1.41329E-05
428	3	5	4	1.35289E-05	1.10276E-05	1.27785E-05	2.03557E-05
411	2	5	1	1.23588E-05	1.16021E-05	1.21318E-05	2.61532E-05
421	3	5	1	1.09636E-05	1.31512E-05	1.16199E-05	2.49274E-05
310	1	4	5	1.48223E-05	1.32757E-05	1.43583E-05	2.41484E-05

5.4.3 Multimodular Neural Networks MMeX

If neural network is used as gating module, it is possible to combine not only outputs of single networks-predictors but to use original data as well. In this way, become not strictly hierarchical multimodular neural network and gating module obtains additional knowledge about how every single predictor processes the data. Multimodular neural networks of this type we will mark as MMeX (Multi Modulat with eXternal inputs).

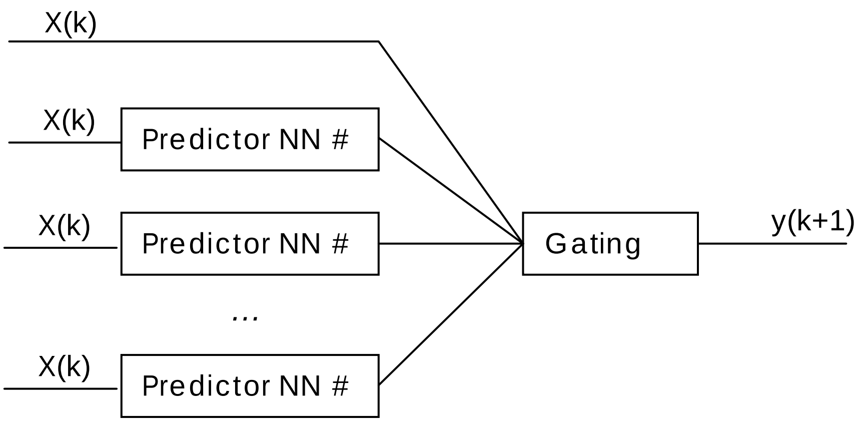


Fig. 19. Architecture multimodular neural network “MMeX” with using original data in gating module

Static Linear Neural Network as gating module of MM multimodular neural network MMeX. Least Mean Squares (LMS) algorithm of weights correction was used for training process. Number of training attempts: 1. Total networks trained for every dynamic process: 1.

Table 24. Results of Linear Neural Network on both sequences

Sequence name	TRAIN	TEST	TRAIN+TEST	VALIDATION
Wolf Numbers	0.001297	0.010035	0.003918459	0.042677144
Mackey-Glass Process	5.83929E-06	8.51037E-06	6.64061E-06	5.97136E-06

Static Multilayered Perceptron as gating module of MM multimodular neural network MMeX. Networks were trained with using Levenberg-Marquardt training algorithm. Maximum number of epochs: 600. Number of neurons in hidden layer, HiddenN: from 1 to 5. Number of training attempts: 2. Total networks were trained for every dynamic process: 10.

Table 25. Best results on sequence “Wolf Numbers”

Gating MLPNN No.	Hidden N	TRAIN	TEST	TRAIN+TEST	VALIDATION
1	1	0.001287	0.010418	0.004026163	0.042971779
2	1	0.001287	0.010418	0.004026192	0.042965025
4	2	0.001197	0.011418	0.004263335	0.040576657
7	4	0.000725	0.015507	0.00515939	0.036101583
5	3	0.000755	0.016766	0.005557926	0.104565267

Table 26. Best results on sequence “Mackey-Glass Process”

Gating MLPNN No.	HiddenN	TRAIN	TEST	TRAIN+TEST	VALIDATIO N
9	5	1.3196E-06	4.48807E-06	2.27014E-06	3.83486E-06
10	5	1.22013E-06	4.61043E-06	2.23722E-06	5.09424E-06
8	4	1.22888E-06	4.70064E-06	2.27041E-06	4.26654E-05
7	4	1.55465E-06	4.97533E-06	2.58085E-06	3.98564E-06
5	3	1.70443E-06	5.07555E-06	2.71577E-06	4.60177E-06

Dynamic Linear Neural Network as gating module of MMeX multimodular neural network. Least Mean Squares (LMS) algorithm of weights correction was used for training process. Tapped delay line length, InputDelayN: from 1 to 10. Number of training attempts: 1. Total networks were trained for every dynamic process: 10.

Table 27. Best results on sequence “Wolf Numbers”

Gating DLNN No.	InputDelay N	TRAIN	TEST	TRAIN+TES T	VALIDATION
1	1	0.00126295	0.01034158	0.003986541	0.043054454
2	2	0.00123581	0.01057764	0.004038358	0.047897703
3	3	0.00117561	0.01142648	0.004250875	0.054009842
4	4	0.00115414	0.01265628	0.004604782	0.063694521
5	5	0.00114432	0.01289156	0.004668492	0.064793899

Table 28. Best results on sequence “Mackey-Glass Process”

Gating DLNN No.	InputDelay N	TRAIN	TEST	TRAIN+TES T	VALIDATIO N
1	1	5.74409E-06	9.12733E-06	6.75906E-06	6.51437E-06
2	2	5.66031E-06	9.58711E-06	6.83835E-06	7.70374E-06
3	3	5.29482E-06	1.04936E-05	6.85446E-06	1.25594E-05
4	4	5.01818E-06	1.15817E-05	6.98722E-06	1.61303E-05
5	5	4.88739E-06	1.27062E-05	7.23304E-06	2.469E-05

Multilayered Perceptron with Tapped Delay Line as gating module of MMeX multimodular neural network. Networks were trained with using Levenberg-Marquardt training algorithm. Maximum number of epochs: 600. Number of neurons in hidden layer, HiddenN: from 1 to 5. Tapped delay line length, InputDelayN: from 1 to 10. Number of training attempts: 2. Total networks were trained for every dynamic process: 100.

Table 29. Best results on sequence “Wolf Numbers”

Gating FTDNN No.	InputDelay N	HiddenN	TRAIN	TEST	TRAIN+TES T	VALIDATIO N
22	1	2	0.00127964 4	0.008606876	0.003477814	0.03780185
23	2	2	0.00143569	0.009494957	0.00385347	0.042929054
2	1	1	0.00156697	0.00953953	0.003958738	0.041137035
1	1	1	0.00156700 7	0.009541525	0.003959363	0.041138109
4	2	1	0.00154653	0.00977659	0.004015548	0.042554942

Table 30. Best results on sequence “Mackey-Glass Process”

Gating FTDNN No.	InputDelayN	HiddenN	TRAIN	TEST	TRAIN+TEST	VALIDATION
82	1	5	1.04589E-06	7.49995E-06	2.98211E-06	4.60201E-06
83	2	5	6.44581E-07	1.43612E-05	4.75956E-06	9.58373E-06
85	3	5	4.59586E-07	2.0337E-05	6.42281E-06	1.72575E-05
42	1	3	7.93076E-06	2.14153E-05	1.19761E-05	3.04193E-05
44	2	3	7.35627E-06	2.49099E-05	1.26224E-05	2.29277E-05
43	2	3	6.43705E-06	3.54946E-05	1.51543E-05	4.99548E-05

Recurrent Nonlinear AutoRegression Neural Network with eXogenous inputs as gating module of MMeX multimodular neural network. Networks were trained with using Levenberg-Marquardt training algorithm and backpropagation through time (BPTT) teqnique. Maximum number of epochs: 30. Number of neurons in hidden layer, HiddenN: from 1 to 5. Tapped delay line length, InputDelayN: from 1 to 5. Number of training attempts: 2. Total networks were trained for every dynamic process: 500.

Table 31. Best results on sequence “Wolf Numbers”

NARXNN No.	InputDelayN	HiddenN	OutputDelayN	TRAIN	TEST	TRAIN+TEST	VALIDATION
210	1	3	5	0.000895341	0.004671681	0.002028243	0.058744786
221	3	3	1	0.00110183	0.007393651	0.002989376	0.041770784
328	3	4	4	0.000534909	0.008188779	0.00283107	0.069641495
101	1	2	1	0.001119768	0.008266566	0.003263807	0.033520944
226	3	3	3	0.001407275	0.008559926	0.00355307	0.046006775

Table 32. Best results on sequence “Mackey-Glass Process”

NARXNN No.	InputDelayN	HiddenN	OutputDelayN	TRAIN	TEST	TRAIN+TEST	VALIDATION
413	2	5	2	1.09349E-05	1.08614E-05	1.09129E-05	1.27645E-05
315	2	4	3	9.93835E-06	1.14634E-05	1.03959E-05	9.96768E-06
403	1	5	2	1.11947E-05	1.18463E-05	1.13902E-05	1.16177E-05
451	6	5	1	1.04818E-05	1.26008E-05	1.11175E-05	1.46583E-05
423	3	5	2	1.22745E-05	1.27749E-05	1.24246E-05	1.41345E-05

5.5 United performance tables for modelling Wolf Numbers and Mackey-Glass Process

For single neural networks, “NN type.” means type of neural network-predictor. In multimodular neural networks, recurrent neural networks were used as networks-predictors only. “NN Type” means type of gating module that makes generalization of predictions of single networks.

Table 33. Performance of best single and multimodular neural networks on sequence “Wolf Numbers”

	NN type	TRAIN	TEST	TRAIN + TEST	VALIDATION
Single Neural Network	Single LNN	0.01134	0.01042	0.01128	0.02276
	Single MLP	0.01063	0.01032	0.01075	0.02468
	Single DLNN	0.02468	0.00592	0.00597	0.01166
	Single FTDNN	0.00204	0.00400	0.00269	0.01181
	Single NARXNN	0.00214	0.00370	0.00297	0.01300
Multimodular Neural Network without using original data in gating module	MM Average	N/A	N/A	0.00262	0.01653
	MM Median	N/A	N/A	0.00231	0.01432
	MM LNN	0.001283	0.008261	0.003376472	0.01930
	MM MLP	0.001279	0.008341	0.003397588	0.01932
	MM DLNN	0.001231597	0.001231	0.00342597	0.02209
	MM FTDNN	0.001580308	0.008151	0.003551544	0.03134
	MM NARXNN	0.001179236	0.006721	0.002841924	0.02498
Multimodular Neural Network with using original data in gating module	MMeX LNN	0.001297	0.010035	0.003918459	0.04267
	MMeX MLP	0.001287	0.010418	0.004026163	0.04297
	MMeX DLNN	0.00126295	0.010341	0.003986541	0.04305
	MMeX FTDNN	0.001279644	0.008606	0.003477814	0.03780
	MMeX NARXNN	0.000895341	0.004671	0.002028243	0.05874

Table 34 Performance of best single and multimodular neural networks on sequence “Mackey-Glass Process”

	NN type	TRAIN	TEST	TRAIN + TEST	VALIDATION
Single Neural Network	Single LNN	0.018738	0.019297	0.018906	0.018598
	Single MLP	0.017549	0.01792	0.01766	0.018866
	Single DLNN	0.00369	0.003339	0.003585	0.0035374
	Single FTDNN	4.13E-06	6.81E-06	4.94E-06	1.27E-05
	Single NARXNN	1.04E-05	7.69E-06	9.62E-06	1.04E-05
Multimodular Neural Network without using original data in gating module	MM Average	N/A	N/A	5.63964E-06	9.12857E-06
	MM Median	N/A	N/A	5.92201E-06	5.81958E-06
	MM LNN	4.45E-06	1.63E-05	8.0028E-06	4.96907E-05
	MM MLP	9.63401E-07	4.99903E-06	2.17409E-06	0.1279415
	MM DLNN	3.85456E-06	1.40081E-05	6.90061E-06	4.25028E-05
	MM FTDNN	6.35888E-07	1.05279E-05	3.60348E-06	1.63934E-05
	MM NARXNN	1.13236E-05	9.84379E-06	1.08797E-05	1.41329E-05
Multimodular Neural	MMeX LNN	5.83929E-06	8.51037E-06	6.64061E-06	5.97136E-06

Network with using original data in gating module	MMeX MLP	1.3196E-06	4.48807E-06	2.27014E-06	3.83486E-06
	MMeX DLNN	5.74409E-06	9.12733E-06	6.75906E-06	6.51437E-06
	MMeX FTDNN	1.04589E-06	7.49995E-06	2.98211E-06	4.60201E-06
	MMeX NARXNN	1.09349E-05	1.08614E-05	1.09129E-05	1.27645E-05

6. Comparison of decomposed, non-decomposed and PID control on the problem of 2D inverse pendulum stabilization for different network sizes

Experiments were performed on the two-dimensional inverse pendulum model that was imported from SIMULINK package. Time that passes between setting the pendulum into equilibrium state and moving it outside the given boundary values due to a high-level noise, was chosen as the control quality estimation.

The goal of the first series of experiments was comparison of the control quality while using single-modular and two-modular neural networks. The model without cart position restrictions was used for this purpose and the only restriction was maximal pendulum angle: the system was set into equilibrium state, then control and noise were applied. Time in seconds, that passes before the pendulum angle exceeded the boundary angle, was used as control quality estimation (in given experiments, there were two series: one with boundary angle equal to approximately 1 degree, and in another it was set to 10 degrees).

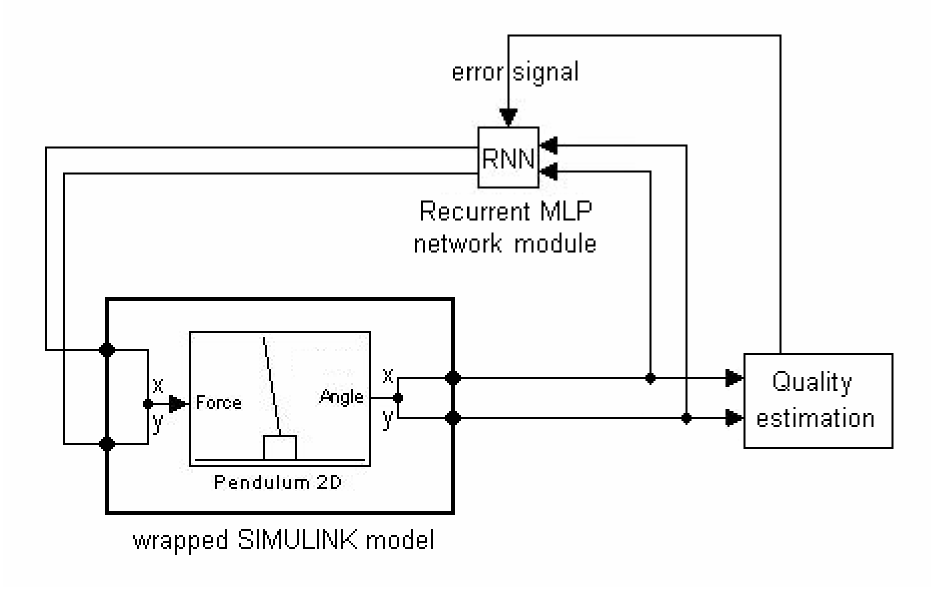


Fig. 20. Control scheme with single-modular network.

Single-modular network received the pendulum's angle and speed of angle change on X and Y axes, and produced control for each axis.
Two-modular network consists of two independent modules, each of them received the pendulum's angle and speed of angle change on either X or Y axis, and produced control for that axis.
In both cases, the reinforced learning method was used. It was built in the following way:
1. The network obtained data on system's state, and produced some control signal U;
2. The control signals U+h and U-h, where h means some small value, were consequently applied to the system;
3. External quality estimation module compared the system's state after applying the first and the second

controls, and decided which of them was better in the means of achieving of the target state;

4. One step of iterative learning was applied for the network, where current system state was used as an input, and target value was set to U+h or U-h (correspondingly to the quality estimation module decision).

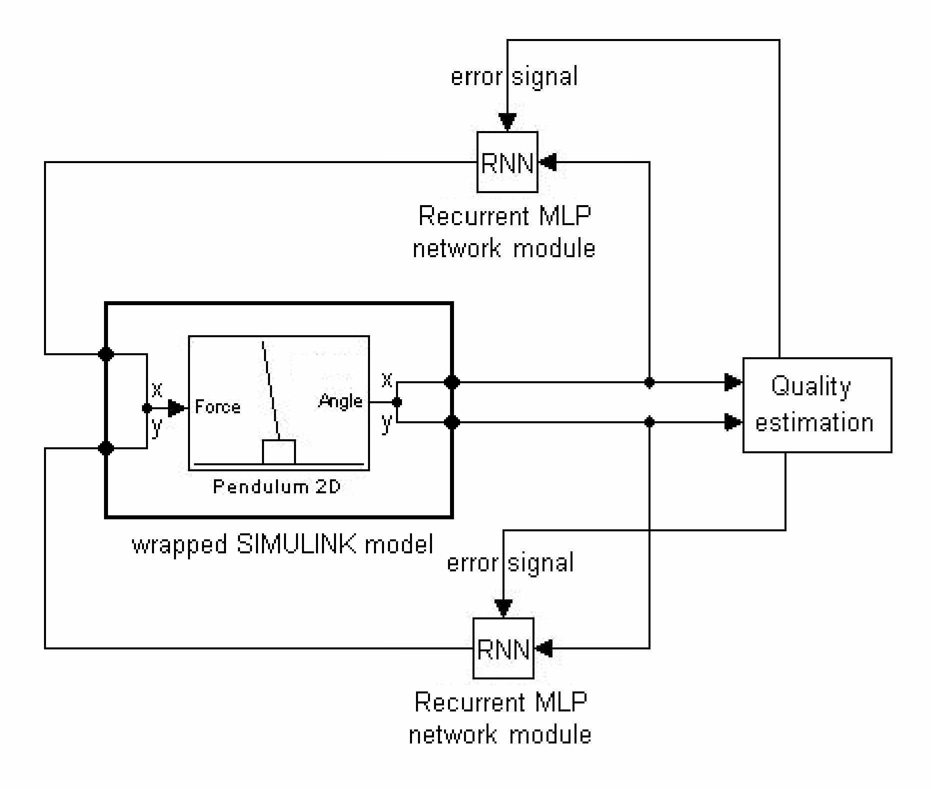


Fig. 21. Control scheme with two-modular network.

As the recurrent network, 3-layer recurrent perceptron, illustrated on Fig. 22, that received 3 delayed inputs. 3 delayed recurrent outputs, and had variable number of hidden neurons, was used. Also there were performed experiments with increased number of recurrent connections, its results weren't significantly different, and experiments with absence of recurrent connections, which results were significantly lower in the terms of control quality.

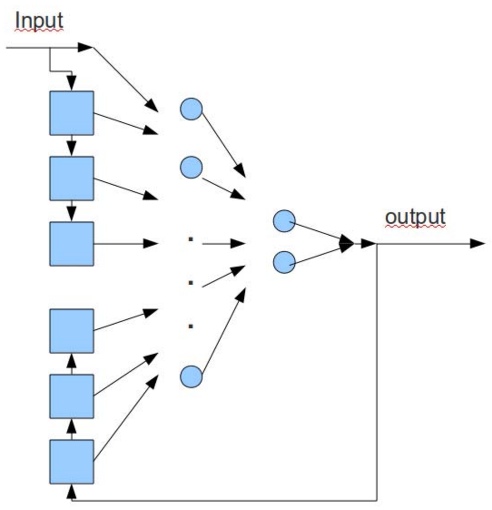


Fig. 22. Recurrent perceptron that was used for experimental study.

The network was trained using recurrent real-time learning method (RTRL).

Experiments were performed for different number of neurons, to study the influence of a network's size on the control quality. Also, as a reference point, the same experiment was performed for PID-controller, with the same level of noise and boundary conditions.
All control schemes were tested in the following way: the pendulum was set into equilibrium state, and then noise and control forces were applied to it. When the pendulum angle exceeded the boundary angle, passed time was fixed, and the process was repeated. As the control quality estimation, the average time for 200 runs was taken.

The following data were obtained in this experiments:

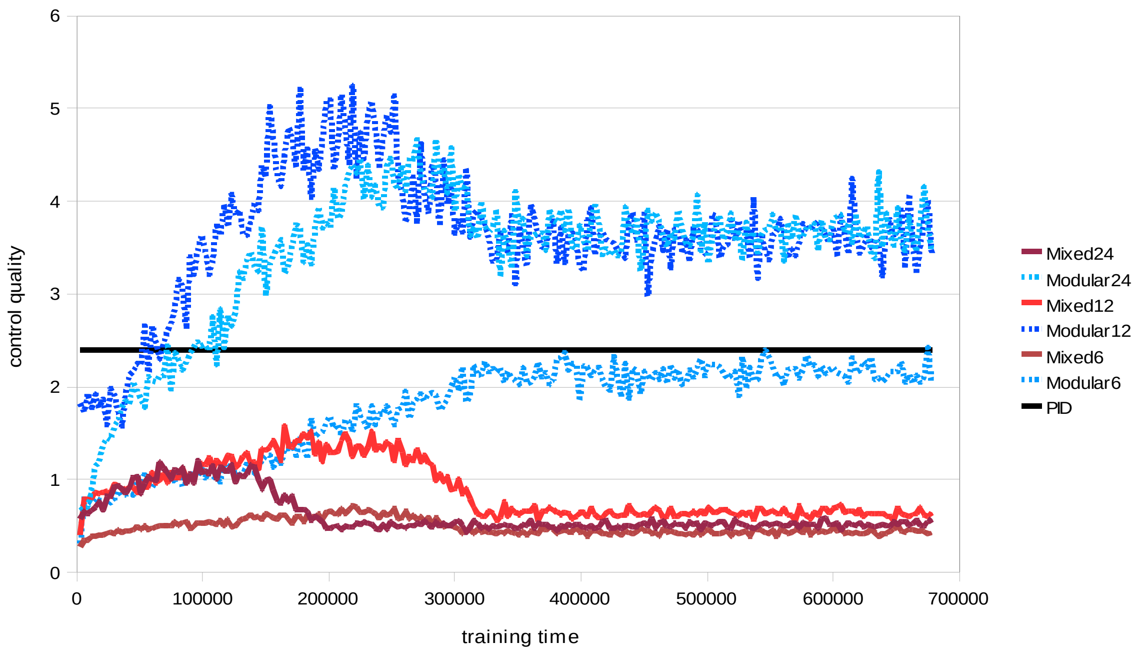


Fig. 23. Dependency of control quality on control scheme, number of neurons and training time. Blue dashed lines correspond to two-modular network, solid red lines correspond to single-modular network, black line corresponds to PID-controller.

From the data, it is clearly seen that two-modular control is significantly better than single-modular for any number of neurons. But still the role of the number of neurons remains unclear in this set of experiments. So another set was made, where two-modular network was tested with different number of neurons in modules, and training/testing process was performed on the same scheme. The maximal achieved control quality, dependent on the number of neurons in the hidden layer, is shown on the Fig. 25.

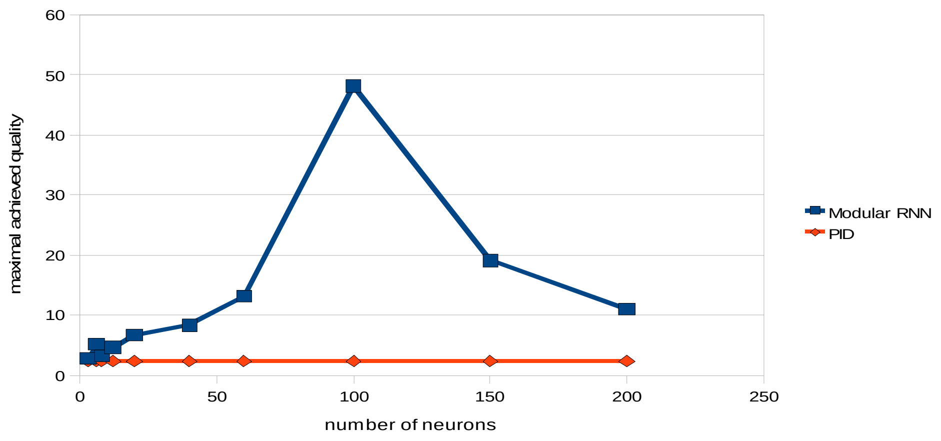


Fig. 24. Dependency of the control quality on the number of neurons in the hidden layer of a two-modular network. For the reference, control quality for PID-controller is shown.

The same experiment was performed for the case, when boundary angle was set to approximately 10 degrees. In this case, advantage of recurrent network on PID-controller was even more significant, and was clearly observed in the case of both two-modular and single-modular networks. This could be explained by the fact, that nonlinear effects are the stronger, the higher is deviation of the pendulum from the equilibrium state, and correspondingly, a nonlinear controller, such as recurrent network, could better fit the controlled object than linear PID-controller.

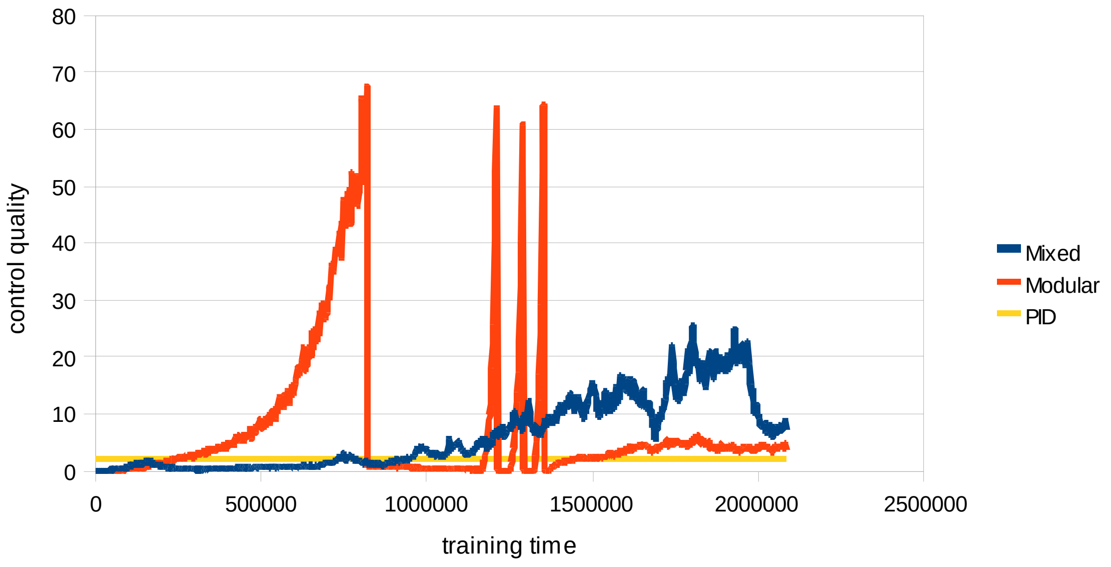


Fig. 25. Dependency of the control quality on control type and training time for the boundary angle set to 10 degrees.

From all figures non-monotonic behavior of dependency of control quality on training time is clearly seen. This is caused by the fact that during training, network's behavior is optimized only locally, while maximization of the control quality according to the chosen criteria requires finding the global optimum. Nevertheless, the chosen training method allows to obtain rather good in global sense networks, if the training process is constantly controlled by testing on independent data.

Also experiments on stabilization of both pendulum's angle and position of the cart, where the pendulum was placed. The main problem is that for the case, when two independent controllers, one for pendulum's angle, and another for cart position, are used, their control signals are contradictory: if we take a linear combination of them, then either the pendulum won't be stabilized, or the cart won't stay in the target state (dependent on the coefficients in the linear combination). Attempt to create a single network, that will keep both pendulum's angle and the cart position, weren't successful – the training process in this case doesn't converge for chosen error calculation method. The problem was solved by decomposing control in the following way: one module was used for pendulum's angle stabilization to the target value, and another module was used for setting the target value for the first module in the way to move the cart into the desired position.

Performed experiments allow to conclude the following:

1 Decomposition significantly increases the control quality;

2 The control quality changes non-monotonic during training of neural networks, so regular testing on the independent data is required;

3 There is an optimal number of neurons, if the number is too low or too high, the control quality is significantly lowered.

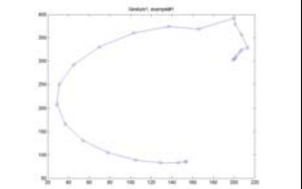
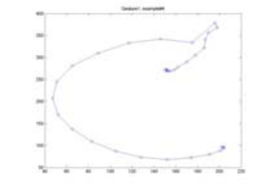
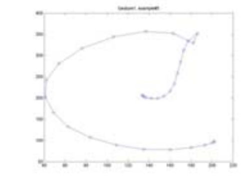
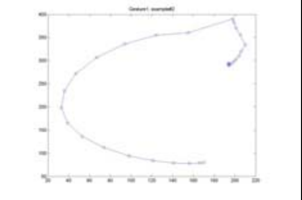
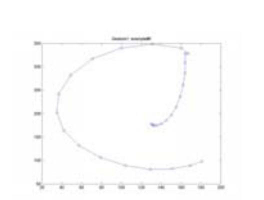
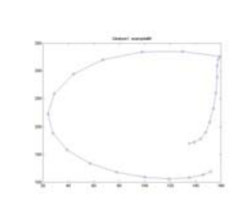
4 Increasing of non-linear effects increases an advantage of neurocontrol on linear control methods.

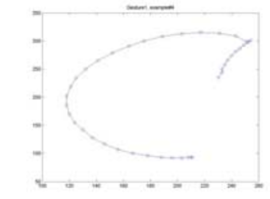
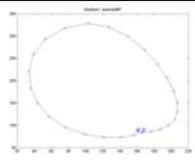
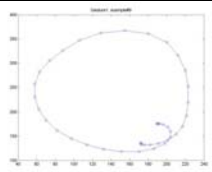
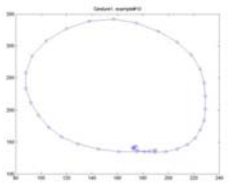
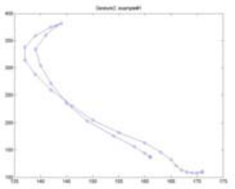
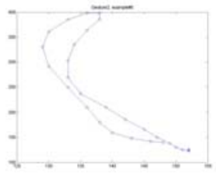
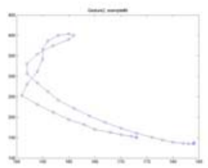
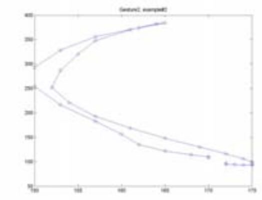
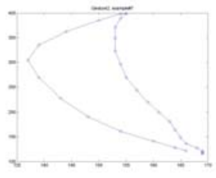
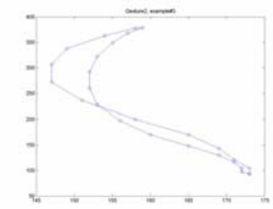
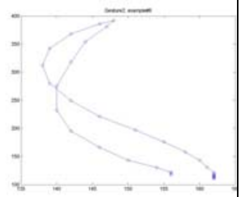
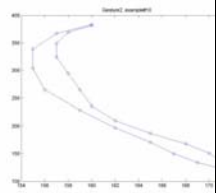
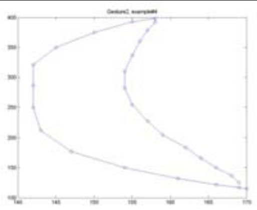
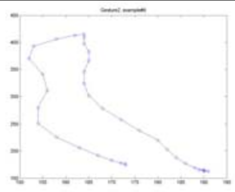
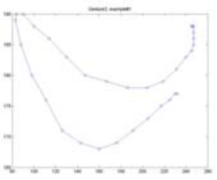
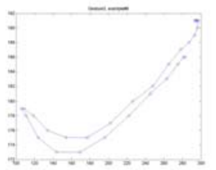
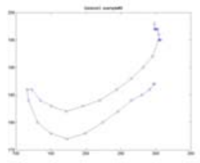
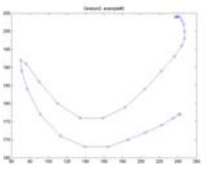
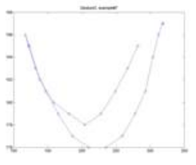
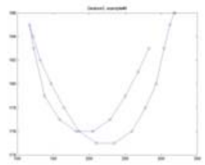
7. **Experiments related to Gesture Recognition**

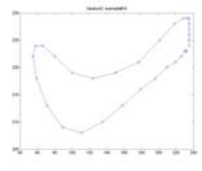
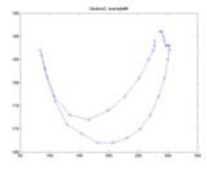
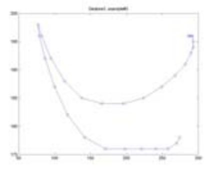
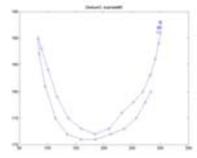
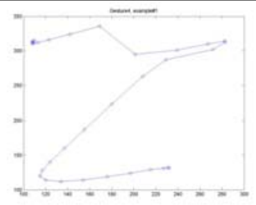
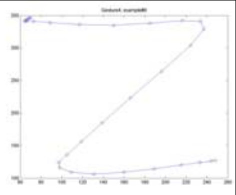
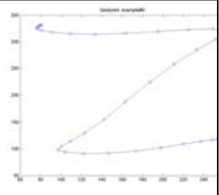
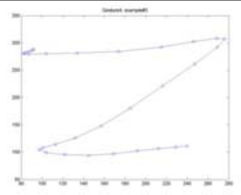
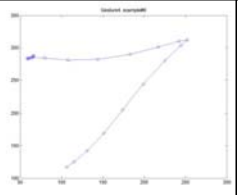
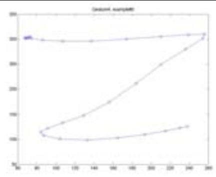
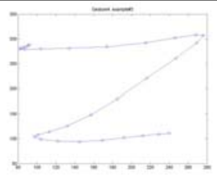
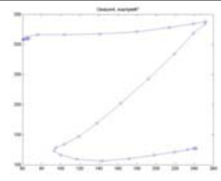
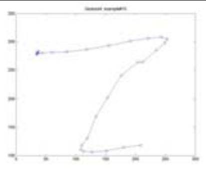
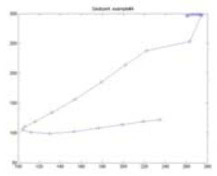
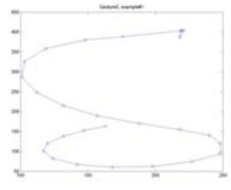
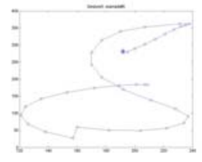
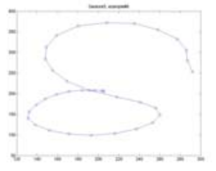
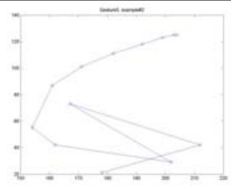
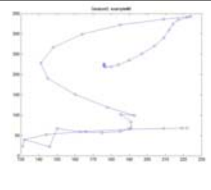
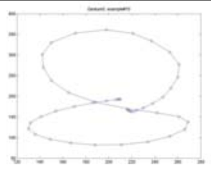
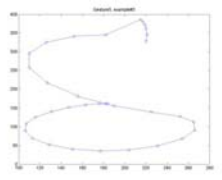
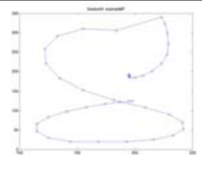
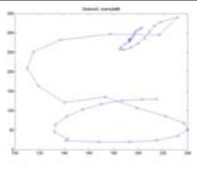
7.1. General description of experiment

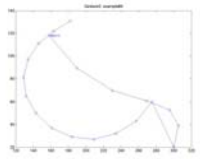
Behavior of dynamic system "Human Gestures" is presented as set of examples of hand movement trajectories in 2D-space. Hand movement is performed by human operator that is located in area of vision of video-camera. Operator hand is dynamic system to model. We use 5 gestures to recognize in our system, they are presented at Table 35. Video camera in real-time broadcasts sequences of images into PC with gesture recognition system installed.

Table 35. Trajectories of gestures for recognition

No	Gesture name	Gesture description	Shapes of obtained from camera trajectories of gestures
1	"Rotation"	Rotation of hand in CW, hand makes full rotation in frontal plane.	     

					
					
2	“Up-down”	Hand movement up and down.			
					
					
					
3	“Left-right”	Hand movement left and right.			
					

					
					
4	"Z"	Hand movement in shape of Z letter.			
					
					
					
5	"8"	Hand movement in shape of 8 letter.			
					
					

Recognition system must return gesture class number by analysis trajectory of hand movement. We suppose that moments when gesture starts and ends are known. Quality of classification is estimated by formula:

$$Error = \frac{N_{CORRECT}}{N} \times 100\%$$

where: $N_{CORRECT}$ – number of correct recognized gestures, N – total number of gestures, given to the system. Dynamic system model “Human Gestures Recognition” described in intermediate report TO3 was used for experiments.

Trajectories for recognition. 50 trajectories (10 trajectories per gesture) that present 2D trajectories of hand movement were recorded for model “Human Gestures Recognition”. They are shown at Table 35. Web-cam A4Tech PK-130MG was used for these purposes. It provides receiving color image sequences in format 640x480 with 22 frames per second. Original preprocessing system for hand detection and extracting center of hand was used. As result, 50 trajectories of point movement that describes movement of hand’s center in 2D-space were obtained. Trajectories that are in fact stretched in time sequences of 2D points have different length, from 30 to 120 points. From 10 sequences for each gesture 5 were marked as sequences for train, 2 – for test, and 3 – for validation.

7.2. Gesture recognition system on the base of single recurrent neural network

Gesture recognition system scheme. Gesture recognition system consists of two modules: “Single NARX NN” and voting module “Voting”.

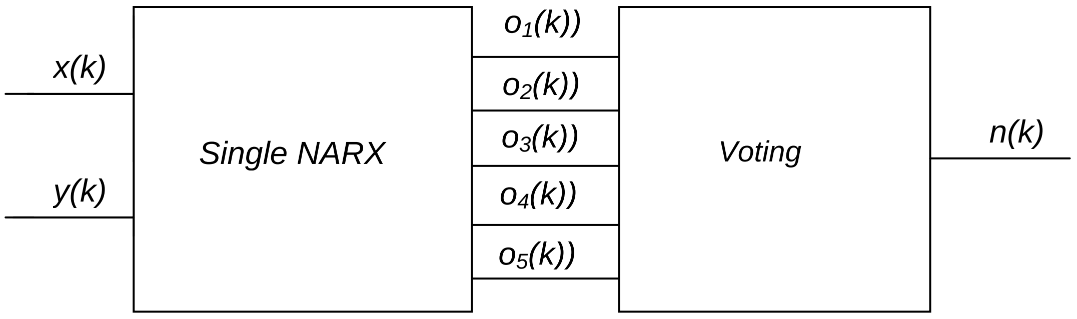


Fig 26. Work scheme of gesture recognition system on the base of one recurrent neural network

Coordinates of object in space $(x(k), y(k))$ in the moment k are comes to recurrent neural network “Single NARX NN” that have as many outputs as gesture classes to be recognized actually are. In ideal case recurrent neural network must generate vector $\bar{o}_I = (1,-1,-1,-1,-1)^T$ for the first gesture, $\bar{o}_{II} = (-1,1,-1,-1,-1)^T$ for the second gesture and so on. But because of unavoidable train errors vector’s components can be not strictly “1” or “-1”. Therefore module “Voting” used rule “the biggest vector component defines the winner”:

$$n = i, if\,[o_i = \max(o_1,...,o_N)]$$

where: n – recognized by system number of gesture class, N – total number of gesture classes, o_i – component of vector of neural network's output, $i = \overline{1, N}$.

Gesture recognition system works in real-time. It receives current values of trajectory's coordinates in 2D $(x(k), y(k))$ every moment k and defines what gesture is responsible to the sequence that comes in to network's input. Final answer is the most frequently appeared gesture number, generated by recurrent neural network. By task definition, neural network doesn't define start and end points of gesture, this work is done by another methods.

Forming train, test and validation sequences. Training, testing and validation of the system were made in off-line. To perform this, tran, test and validation data sequences were formed from sequentially joining trajectories of hand movement presented in table on Table 35. First, united sequences "P" and "T" were formed. Sequence "P" includes 2D-trajectories that enters to neural network. It was formed by sequential joining points of all gesture trajectories. Length of train part of the sequence was 1302 points, test part consists of 724 points, validation part consist of 1677 points. In these subsequences 5, 2 and 3 gestures for each of 5 gesture classes as 2D-trajectories were coded. Target sequence "T" behavior of which recurrent neural network must mimic was formed artificially. It's length is identical to length of input vector "P" and it consists vectors that codes relation of corresponding points in sequence "P" to some gesture. Vectors have 5 components, "1" if number of component is the same as number of gesture is in sequence "P" and "-1" otherway.

7.3. Experiment results for gesture recognition system on the base of single recurrent neural network

For recognition of gestures dynamic recurrent multilayered perceptron with tapped delay line NARXNN was used. Structure of this neural network is similar to multilayered perceptron with tapped delay line but it additionally has recurrent feedbacks that connects output of network with input through additional delay line. Number of delays in this line are marked as *OutputDelayN*. Parameters *InputDelayN* and *HiddenN* means number of delays in input line and number of neurons in hidden layer. Training the neural network was performed by BPTT method with using Levenberg-Marquardt training algorithm. Experiments were done with different values of neural network parameters: number of delays in input tapped delay line: InputDelayN: 1, 11 and 21; number of neurons in hidden layer HiddenN: 1, 6, 11, 16 та 21; number of delays in output tapped delay line OutputDelayN: 1, 11 and 21; number of training seances for each architecture: 2. Total networks were trained: 90. On Fig. 27– 28 output values of trained recurrent neural network output (thin red line) and target values for network to mimic (blue line).

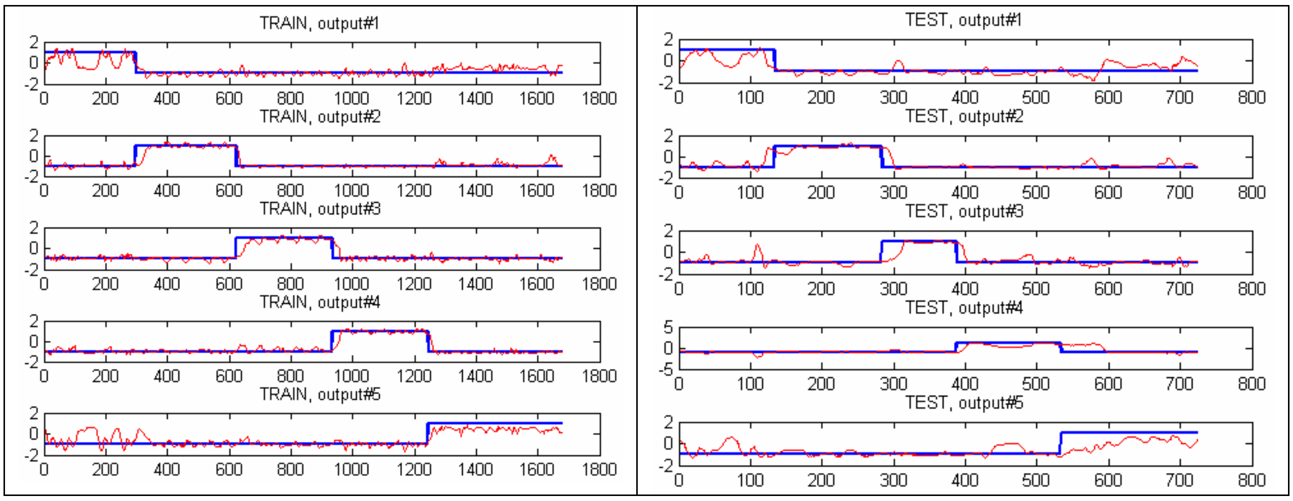


Fig. 27. Output of network "Single NARX NN" on train sequence (left) and test sequence (right)

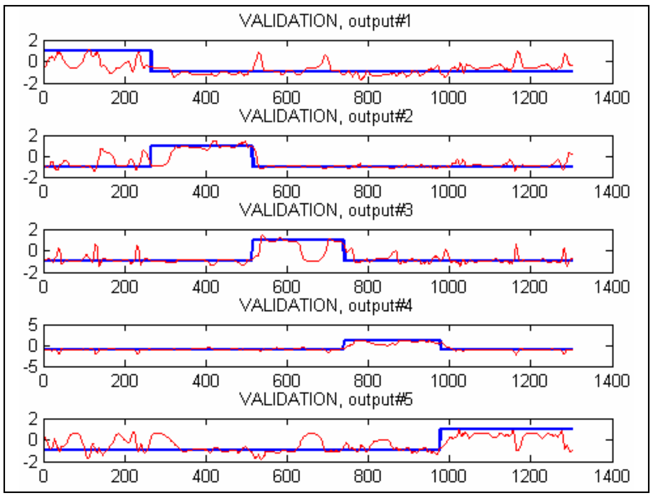


Fig. 28. Output of recurrent neural network ”Single NARX NN” on validation sequence

Table 36. Mean squared error values of single recurrent neural networks

Network No.	Input DelayN	Hidden N	Output DelayN	TRAIN	TEST	TRAIN+TEST	VALIDATION
86	21	21	1	0.166772	0.311923	0.268154196	0.361011541
68	21	16	1	0.115909	0.381454	0.301381029	0.566969266
43	11	11	1	0.244064	0.402316	0.354596667	0.561673392
85	21	21	1	0.04702	0.406496	0.29809941	0.42239987
26	11	6	1	0.359287	0.407681	0.393088158	0.539329424

Classification accuracy at the output of “Voting” module for best neural network: 96% on TRAIN sequence, 90% - on TEST sequence and 93% on VALIDATION sequence.

7.4 Gesture recognition system on the base of multimodular recurrent neural network

Gesture recognition system scheme. Multimodular system has few independent recurrent neural networks, each of them was trained to recognize one gesture only. In recognition regime these neural networks works in parallel mode and generates “1” on their own gesture and “-1” on other gestures. Final decision was done by analytic module “Voting” that worked by the same formula as gesture recognition system based on single recurrent neural network described in section 7.2.

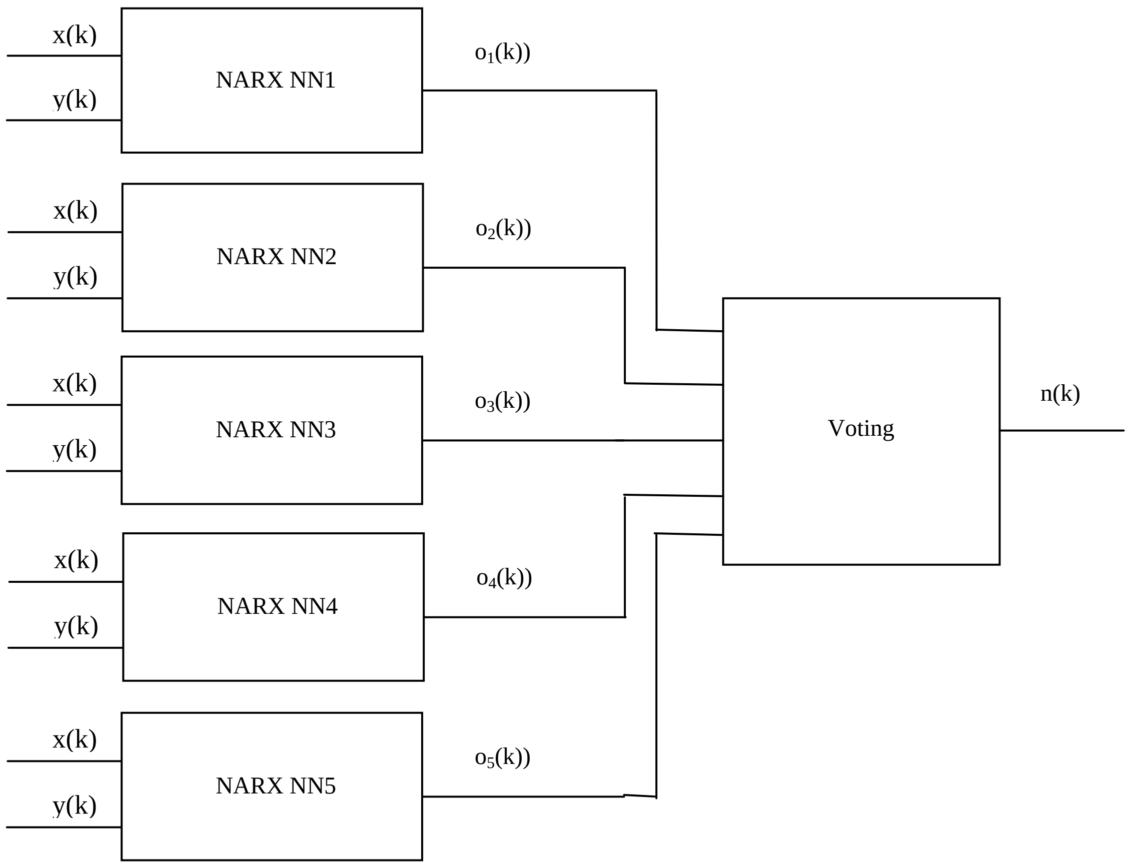


Fig 29. Gesture recognition system on the base of multimodular recurrent neural network

Forming train, test and validation sequences. Input sequences for multimodular neural network are the same one as for single neural network presented in previous paragraph. During the training, testing and validation all neural networks receives the same data. Input sequences for multimodular recurrent neural network are the same as for single recurrent neural network. Otherway, target sequences are one-dimensional and unique for every neural network NARX NN1 – NARX NN5 and are formed individually. For every neural network estimated reaction has value “1” for area of input data that belongs to appropriate to this concrete neural network part of gesture’s trajectory, and “-1” for other areas.

Experiment results. Recurrent neural networks NARXNN were used as single predictors NN1—NN5 of multimodular neural network gesture recognition system. They were the same type of gesture recognition system based on single recurrent neural network. Training of networks was performed with the same parameter values. Training was made my BPTT method with using Levenberg-Marquardt algorithm. The following parameter values were used: number of delays in input tapped delay line: InputDelayN: 1, 11 and 21; number of neurons in hidden layer HiddenN: 1, 6, 11, 16 та 21; number of delays in output tapped delay line OutputDelayN: 1, 11 and 21; number of training seances for each architecture: 2. Total networks were trained: 5x90=450. On Fig. 30– 31 outputs of trained networks NARX NN1—NN5 (thin red line) and target values (blue line) are presented.

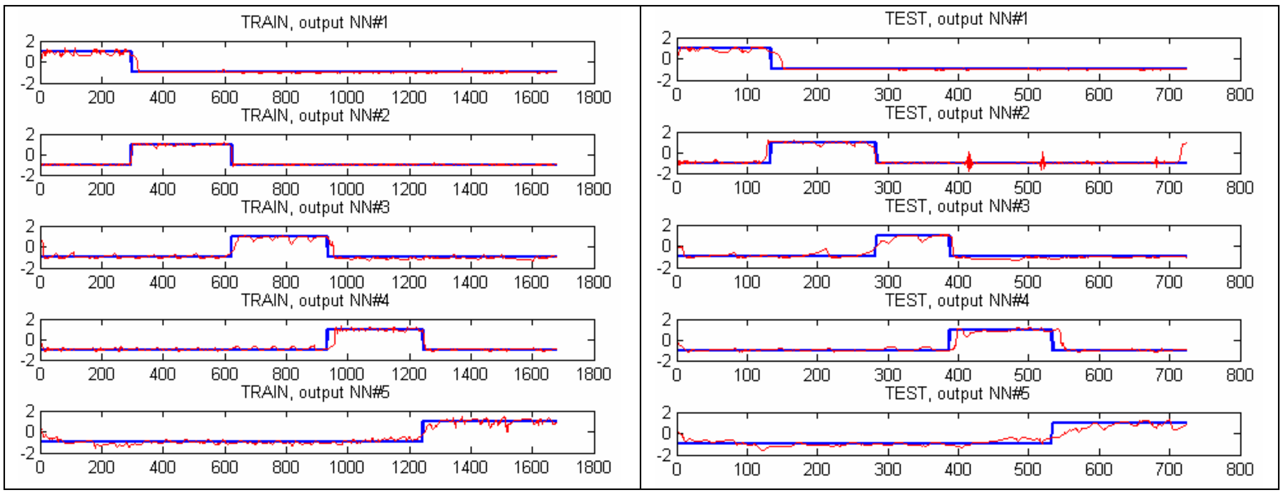


Fig. 30. Outputs of best networks NARX NN1—NN5 of multimodular neural network on train sequence (left) and test sequence (right)

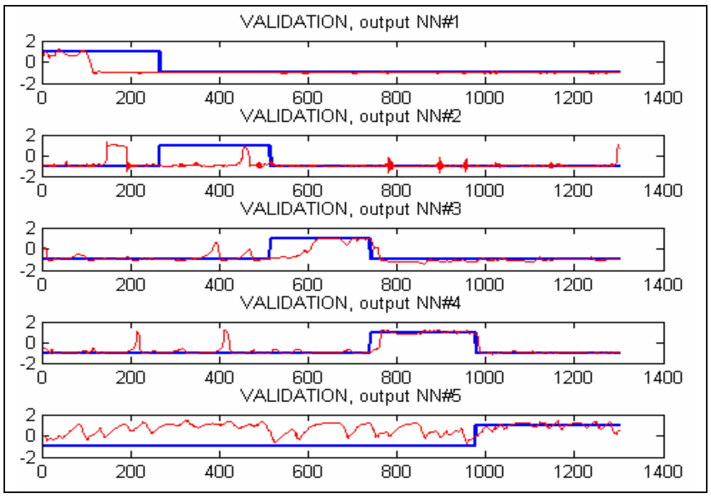


Fig. 31. Outputs of best networks NARX NN1—NN5 of multimodular neural network on validation sequence

Table 37. Mean squared error values of single predictors NARX NN1 — NN5 of multimodular recurrent neural network

Network No.	Input DelayN	Hidden N	Output DelayN	TRAIN	TEST	TRAIN+TEST	VALIDATION
NN1-73	1	21	1	0.040767525	0.05493745	0.050664634	0.479170705
NN1-18	21	1	21	0.002327401	0.05924862	0.042084536	2.009426134
NN1-79	11	21	1	0.007140702	0.06982852	0.05092557	0.386310225
NN1-17	21	1	21	0.003047696	0.0719962	0.051205396	2.008662652
NN1-44	11	11	1	0.008633063	0.07446668	0.054615145	0.381062496
NN2-86	21	21	1	0.005307402	0.09537677	0.068217162	0.846036657
NN2-73	1	21	1	0.023668605	0.11596472	0.088133656	0.373931189
NN2-37	1	11	1	0.075512629	0.13767245	0.118928712	0.226692226
NN2-44	11	11	1	0.057649858	0.15595118	0.126309296	0.139544598
NN2-55	1	16	1	0.022962567	0.16552161	0.122534209	0.50351598
NN3-20	1	6	1	0.072230793	0.07063831	0.07111851	0.281257422
NN3-25	11	6	1	0.061806159	0.12364241	0.104996241	0.428531122
NN3-73	1	21	1	0.083221583	0.12731249	0.114017275	0.420151395
NN3-74	1	21	1	0.097255481	0.13582207	0.124192659	0.124973515
NN3-42	1	11	21	0.101425867	0.1386366	0.127416039	0.366617872
NN4-55	1	16	1	0.064420044	0.12721362	0.108278782	0.141588642
NN4-19	1	6	1	0.081103873	0.13021424	0.11540545	0.100656317
NN4-70	21	16	11	0.042260174	0.13050101	0.103892777	0.115017848
NN4-72	21	16	21	0.028281518	0.14946179	0.112920967	0.64684883
NN4-74	1	21	1	0.085536645	0.17051446	0.144890166	0.141551221
NN5-64	11	16	11	0.062958725	0.13926395	0.116254793	2.152204312
NN5-73	1	21	1	0.050744392	0.21335585	0.164321825	0.108704296
NN5-54	21	11	21	0.061317244	0.22066651	0.172616169	1.762151972
NN5-53	21	11	21	0.006314346	0.23750186	0.167789343	0.542285809
NN5-71	21	16	21	0.068937353	0.2502816	0.195598871	0.116428196

Classification accuracy at the output of "Voting" module for best set of neural networks NARX NN1 – NN5 of this multimodular neural network was: 100% on TRAIN sequence, 100% - on TEST sequence and 46% on VALIDATION sequence.

Comparatively low classification accuracy on VALIDATION sequence can be explained by high specialization of modules with using selected scheme of organization of multimodular neural system. Therefore, the most promising solution for this class of problems is multimodular architecture that includes few single neural networks and additional recurrent neural network that works as referee instead of currently used "Voting" module.

Results and Discussion

Main results of this project are:

1. New approach to the problem of high computational complexity of the training process of recurrent neural networks was proposed:
   - Theoretical base of Open Recurrent Neural Network was developed.
   - New non-iterative method for training of Open Recurrent Neural Network was designed.
   - New type of Dynamic Associative Memory was developed and its experimental model was tested.
2. Methods of neurocontrol task decomposition for recurrent MLP networks and Dynamic Associative Memory were developed:
   - Methods for decomposition of neurocontrol task by principles “decomposition of controlled object” and “committees of experts” were proposed and implemented.
   - Model of inverse neuroemulator on the base of Dynamic Associative Memory was developed.
   - Methods of multimodular neurocontrol were experimentally studied on 1D and 2D dynamic objects. Estimations of their efficiency were obtained.
3. Experimental base for research of multimodular neurocontrol neural networks and neurocontrol systems was developed. It includes:
   - Software experimental tool “Neuroconveyor” containing multimodular recurrent neural networks.
   - Tools for providing research and development related to neurocontrol.
   - Open library of dynamic processes.

Considerable amount of theoretical and experimental results was achieved during work on this project. These results can be used in applied research and new theoretical research in neurocontrol area. Most promising are the following directions of future research:

1. <u>Neural Dynamic Associative Memory for applied intelligent systems.</u>
   Dynamic Associative Memory that was developed during this project needs for future improvement and development of the technology for its applied use.
2. <u>Problem of stability of inverse models in neurocontrol.</u>
   Reason of instability of dynamic object’s control system could be interrupting principle of causality during modeling the inverse dynamics of object. For the needs of neurocontrol object’s model is obtained by training the neural network, its inverse dynamics is modeled by applying the error backpropagation mechanism to train a feedforward network. Careless use of this mechanism could lead to hidden causality interruption. This leads to inadequate object’s model, and is dangerous in the case of adaptive control, when the model is constantly updating. This threat becomes even higher when multimodular approach that allows concurrency between modules is applied. Stability of neurocontrol is a fundamental problem and its studying is important even for interpretation of complex economic and social processes.
3. <u>Neural Dynamic Associative Memory as an analogue of neural mechanisms of human brain.</u>
   Neural Dynamic Associative Memory can be considered as an analogue of mechanisms of brain memory that explains processes of forming the tracks of neural activity and fact of renewal of memory contents after deceases. It completes model that was proposed in our previous research works: A.M. Reznik, A.S. Sitchov, O.K. Dekhtyarenko, and D.W. Nowicki, Associative Memories with "Killed" Neurons: the Methods of Recovery // Proc. of the International Joint Conference on Neural Networks, July 20-24, 2003. Portland, Oregon; Reznik A.M. Hopfield ensembles in lateral neurostructures of the cerebral cortex // Mathematical Machines and Systems. - 2006.- N1. P. 3-12. (on Russian).

The most important applied result of the project is developing of software system “Neuroconveyor” that was used for performing of complicated experiments on neurocontrol in real-time. Part of experiments was done in package MATLAB + SIMULINK by MathWorks. MATLAB system has advanced and well-developed toolbox for work with artificial neural networks. This allowed us to test many variants of different recurrent and non-recurrent neural networks for solving task of modeling dynamic processes presented as value sequences (Wolf Numbers and Mackey-Glass Process) and Hand Gesture Recognition task. However, SIMULINK shell that was created especially for modeling dynamic control systems has significant disadvantages. In the last to the date version of MATLAB 7.10.0.499 (R2010a) the following problems were found:

1) SIMULINK doesn’t allow to train neural networks for controlling the dynamic object in real-time. Using neural networks that were previously trained in MATLAB workspace is possible only. That makes impossible using broad class of real-time training algorithms such as RTRL and Temporal-Difference Learning.

2) MATLAB command for exporting trained neural networks from MATLAB workspace to SIMULINK “gensim” contains error that causes some sub-blocks of generated neural network to have one tact length 1 sec instead of really used tact length in concrete dynamic object’s numerical model that is usually less than 0.01 sec. These parameters have to be fixed manually every time after neural network is trained and it is tedious process.

3) ”Control systems” section of ”Neural Networks Toolbox” contains only three types of neurocontrol systems: ”NN Predictive Control”, “NARMA-L2”, “Model Reference Control”. These systems are implemented as black-box models, they are impossible to modify. For example, it is impossible to replace the single neural networks with mutimodular neural networks for them.

4) ”Control systems” section of ”Neural Networks Toolbox” doesn’t contain very popular nowadays neurocontrol system scheme “neurocontroller + direct neuroemulator that works through backpropagating the error” that is also known as “backpropagation through time scheme” [11]. Such systems have to be modeled in comparative motives and it is complex task.

Developed during this project software system “Neuroconveyor” was designed especially for work with dynamic neural systems and therefore it has no disadvantages mentioned above. Interfaces of this system are designed especially for providing the neurocontrol experiments software and allows to import and use models of dynamic systems and functional blocks of SIMULINK that makes it more powerful. Such import was provided during experiments with model of inverted pendulum.

References

[1] A.M. Reznik Dynamical Recurrent Neural Networks // Mathematical Machines and Systems, №2, 2009. p.3-26. (on Ukrainian).
[2] A.M. Reznik, D.A. Dziuba Dynamic Associative Memory Based on Open Recurrent Neural Network// Proceeding of IJCNN’09, Atlanta, Georgia, USA, june 14-19, 2009.
[3] A.M.Reznik, D.A. Dziuba Dynamic Associative Memory Based on Open Recurrent Neural Network // Mathematical Machines and Systems, №2, 2010. p.45-51. (on Ukrainian).
[4] Simon Haykin, “Neural Networks: A Comprehensive Foundation” 2nd Ed., 1999 Prentice-Hall, p. 126.
[5] Simon Haykin, “Neural Networks: A Comprehensive Foundation” 2nd Ed., 1999 Prentice-Hall, p. 156.
[6] Simon Haykin, “Neural Networks: A Comprehensive Foundation” 2nd Ed., 1999 Prentice-Hall, p. 644.
[7] Simon Haykin, “Neural Networks: A Comprehensive Foundation” 2nd Ed., 1999 Prentice-Hall, p. 733.
[8] A. J. Sharkey, “Modularity, combining and artificial neural nets”, Connection Science, 9(1):3–10, 1997.
[9] R. Avnimelech, N. Intrator, “Boosting Regression Estimators”, Neural Computation 1999, Vol. 1, Issue 2.
[10] Robert E. Schapire, “The Boosting Approach to Machine Learning: An Overview”, MSRI Workshop on Nonlinear Estimation and Classification, 2002.
[11] Sigeru Omatu, Marzuki Khalid, Rubiyah Yusof Neuro-Control and its Applications/ Springer-Verlag, London Limited, 1996.

Summary of personal commitment

In project took part:
- A. Reznik – project manager (sections 1, 2 of the final report).
- A. Chernodub – responsible for the second research direction (sections 3, 5, 7 of the final report).

- D. Dziuba – responsible for the third research direction (sections 4, 6 of the final report).

Description of travels

D. Dziuba had travel to USA at 12-20 June 2009. He took part in conference IJCNN’09, Atlanta, Georgia, USA, June 14-19, 2009, where work related to the project was presented: A.M. Reznik, D.A. Dziuba „Dynamic Associative Memory Based on Open Recurrent Neural Network”.

Information about major equipment and materials acquired, other direct costs, related to the project

We planned to buy two notebooks, projector and consumables with total cost $2000, but because time delay of start date of this project and our participation in International conference this activity was moved to the second quarter. The following equipment was bought:

1.Notebook Lenovo IBM 3000 G550-4Aplus Black (59-022219)	4801.10 UAH.
2. Notebook Lenovo IBM 3000 G550-4L-1 Black (59-023919)	3960.35 UAH.
3. Projector AcerX1230K(EY.J9805.001)	4402.85 UAH.
Total, without VAT	13164.30 UAH.
Pure VAT	2632.86 UAH.
Total with VAT	15797.16 UAH.

This equipment is situated in Neurotechnologies Dept. of IMMSP NASU and is used by theme of project P-357.

Table Redirection

Reference documents &date (1)	New requested category, or old category with new cost (2)	Requested cost (new) (3)	Original (old) category (4)	Estimated cost (old) (5)	Redirected cost (6) old – new
Quarter <01>					
L01, 5.06.2009	**4-Equ. 4. Equipment.** 4.9 Laptop	0	**4-Equ. 4. Equipment.** 4.9 Laptop	700	-700
L01, 5.06.2009	**4-Equ. 4. Equipment.** 4.10 Laptop	0	**4-Equ. 4. Equipment.** 4.10 Laptop	700	-700
L01, 5.06.2009	**4-Equ. 4. Equipment.** 4.11 Video projector	0	**4-Equ. 4. Equipment.** 4.11 Video projector	500	-500
L01, 5.06.2009	**5-OS. 4.Office supplies.** 4.1. Laptop peripherals	0	**5-OS. Office supplies.** 4.1. Laptop peripherals	100	-100
05.06.09	**7-Trav. 5.International.** 5.6 USA, Atlanta, Georgia, IJCNN’09 one person	3600	**7-Trav. 5.International.** 5.6 USA, Atlanta, Georgia, IJCNN’09 one person	0	+3600
Total by L01					+1600
Quarter <02>					
	GRANT_NFWS No 21472 Dziuba Dmitry	1152	GRANT_NFWS No 21472 Dziuba Dmitry -	924	+224
	GRANT_NFWS No 21473 Chernodub Artem	1152	GRANT_NFWS No 21473 Chernodub Artem	924	+224
			Subtotal		+448
	TRAVEL_OUT. No 5220 USA, Atlanta, Dziuba Dmitry	3,181.26	TRAVEL_OUT. No 5220 USA, Atlanta, one person	3,600.00	-418,74
	Equipment-non-capital No537621, No537622, No537624	1,950.27	Equipment-non-capital No537621, No537622, No537624	1900	+50.27
	MATERIALS No5205	0	MATERIALS No5205	100.00	-100.00
Total by L02					-20.47